Systèmes d'Information & Base de Données

A. BENMAKHLOUF

Site Web: www.cours-informatique.be.ma

Systèmes d'Information & Base de Données

- Chapitre-1 : Base de Données Relationnelle - Modèle Entité/Association (E/A)
- Chapitre-2 : Modèle Relationnel (MR) & Algèbre Relationnel
- Chapitre-3 langages d'interrogation et de manipulation de données SQL

Objectifs

- Avoir la possibilité de modéliser un problème issu d'une situation réelle afin d'en déduire une structure de base de données

- Donner les savoir-faire nécessaires pour réussir à gérer et manipuler les bases de données.

Chapitre-1
Base de Données Relationnelle - Modèle Entité/Association E/A

Base de données & SGBD

Une Base De Données (BDD) est un **ensemble structuré de données** enregistrées sur des supports informatique pour satisfaire simultanément plusieurs utilisateurs de façon sélective. Cette BDD est gérée par un SGBD

Un Système de Gestion de Bases de données (SGBD) est un outil (logiciel) permettant de gérer et de manipuler des BDD.

Objectifs des SGBD :

- Exploitation de gros volumes de données avec une structuration des données et méthodes d'accès efficaces,
- Exploitation par différents types d'utilisateurs avec des différents outils d'accès ou interfaces-utilisateurs.
- Gestion de données sensibles avec sécurité et fiabilité maximal.

Base de données

Mauvaise représentation des données et Problématique

Nom Client	Adresse	Ville	N°Commande	Date-commande	Désignation	Quantité	Prix
Nom1	Adresse1	Rabat	C1	18/10/2012	Article1	2	233,00
Nom1	Adresse1	Rabat	C1	18/10/2012	Article2	5	24,00
Nom1	Adresse1	Rabat	C1	18/10/2013	Article3	6	300,00
Nom1	Adresse1	Rabat	C2	19/10/2014	Article1	5	233,00
Nom1	Adresse1	Rabat	C2	19/10/2014	Article2	5	24,00
Nom2	Adresse3	Fès	C3	20/10/2014	Article1	5	233,00
Nom2	Adresse3	Fès	C3	20/10/2014	Article2	8	24,00
Nom2	Adresse3	Fès	C4	21/10/2014	Article1	2	233,00
Nom2	Adresse3	Fès	C4	21/10/2014	Article3	5	300,00

- Anomalies lors d'une insertion
- Anomalies lors d'une modification
- Anomalies lors d'une destruction

Bonne représentation des données

Une bonne présentation est celle qui :



Etre capable de représenter individuellement les clients les commandes et les produits, de manière à ce qu'une action sur l'un n'entraîne pas systématiquement une action sur l'autre.



Définir une méthode *d'identification* d'un client d'une commande et d'un produit, qui permette d'assurer que la même information est représentée une seule fois .



Préserver le lien entre les clients, les commandes et les produits, mais sans introduire de redondance.

Bonne représentation des données

Id-client	Nom Client	Adresse	Ville
CL1	Nom1	Adresse1	Rabat
CL2	Nom2	Adresse3	Fès

N° article	Désignation	Prix
A1	Article1	233
A2	Article2	24
A3	Article3	300

N°Commande	Date-commande	Id-client
C1	18/10/2011	CL1
C2	19/10/2011	CL1
C3	20/10/2011	CL2
C4	21/10/2011	CL2

N°Commande	N° article	Quantité
C1	A1	2
C1	A2	5
C1	A3	6
C2	A1	5
C2	A2	5
C3	A1	5
C3	A2	8
C4	A1	2
C4	A3	5

Reconstitution intégrale de l'information décomposée

Conception d’un schéma relationnel (Modèle Entité/Association)

Le Modèle

Le modèle **E/A** (Modèle Conceptuel de Données), conçu en 1976, est à la base de la plupart des méthodes de conception.

Modèle de données –> **décrire la réalité perçue à travers les données mises en jeu** (indépendamment des opérations que l’on effectuera ultérieurement dessus)

But: fournir des outils et un cadre rigoureux pour l’analyse des données et de leurs liaisons.

Concepts de base: entité, association, propriété (attribut) et valeur (et type de valeur).

Modèle Conceptuel de Données (MCD) – Modèle E/A

Il représente une description abstraite et détaillée de l’ensemble des objets, ou de concepts concernés par les traitements. sans réfléchir aux possibilités de traitement par l’ordinateur.

Il a pour but d'écrire de façon formelle les données qui seront utilisées par le système d'information. Il s'agit donc d'une représentation des données, facilement compréhensible.

Nous étudierons le modèle basé sur le couple **Entité - Association**

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

Approche par l'Exemple

- Le client « client1 » a passé la commande C1 contenant les produits P1 et P2
- Le même client « client1 » a passé la commande C2 contenant les produits P2 et P3
- Le client « client2 » a passé la commande C3 contenant les produits P1 et P2
- Le même client « client2 » a passé la commande C4 contenant les produits P2 et P3
- La commande C1 a donné lieu à la facture F1
- La commande C2 a donné lieu à la facture F2
- La commande C3 a donné lieu à la facture F3
- La commande C4 a donné lieu à la facture F4

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

La description de ce réel peut aussi se résumer par le tableau suivant:

Client	Commande	Produit	Facture
Client1	C1	P1, P2	F1
	C2	P2, P3	F2
Client2	C3	P1, P2	F3
	C4	P2, P3	F4



4 ensembles (Client; Commande; Produit et Facture) Appelés **ENTITE**

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

Définition

On désigne par entité *tout objet* que l'on peut décrire. Il doit être *identifiable*, *pertinent pour l'application* et concerné *par les règles de gestion.*

Formalisme

Nom de l'entité

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

Propriété d'une entité (Attribut)

La propriété représente **la plus petite quantité** d'informations **élémentaire** caractérisant une entité, que l'on peut utiliser d'une manière **autonome** **et** qui **présente un intérêt** pour le domaine étudié

: Information pertinente

: ne figure pas dans d'autre entité

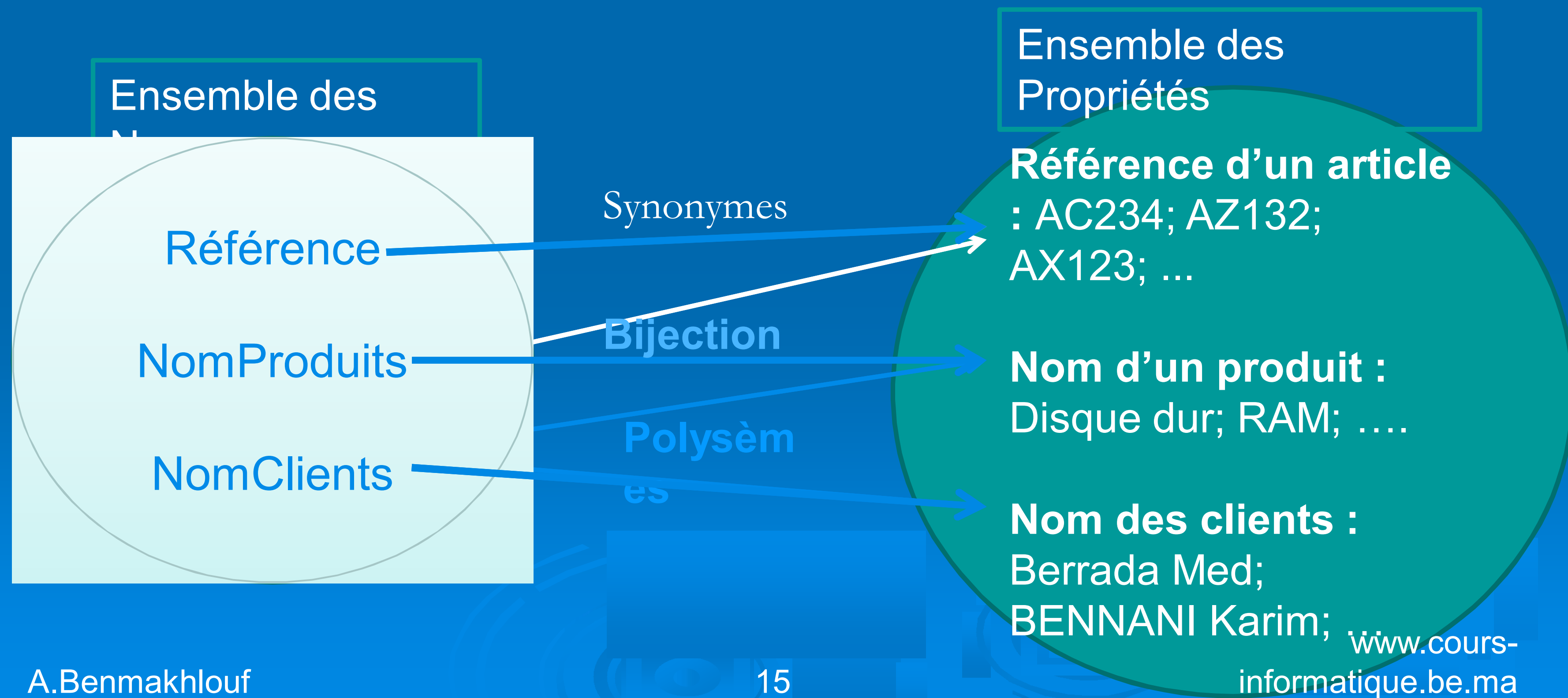
: non déductible d'autres propriétés

: (Atomique) chacune des valeurs qu'elle regroupe n'est pas décomposable. (cette décomposition est relative à son exploitation dans le SI)

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

L'identification des propriétés se fait par des **noms explicites**, qui présentent un intérêt pour le domaine à étudier. Cette identification doit garantir **une bijection** entre l'ensemble des noms et l'ensemble des propriétés à gérer.



Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

Exemple : L'entité **client** peut avoir les propriétés suivantes : **Numéro client**, **Nom ou raison sociale**, **Adresse**, **Ville**, **Fonction**, etc.

Clients
Num client
Société
Contact
Fonction
Adresse
Ville

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

Occurrence d'une propriété. Toute valeur prise par la propriété définit une occurrence de la propriété.

Occurrence d'une entité. L'ensemble de valeurs prises par les propriétés caractérisant une entité définit l'occurrence d'une entité.

Modèle Conceptuel de Données (MCD)

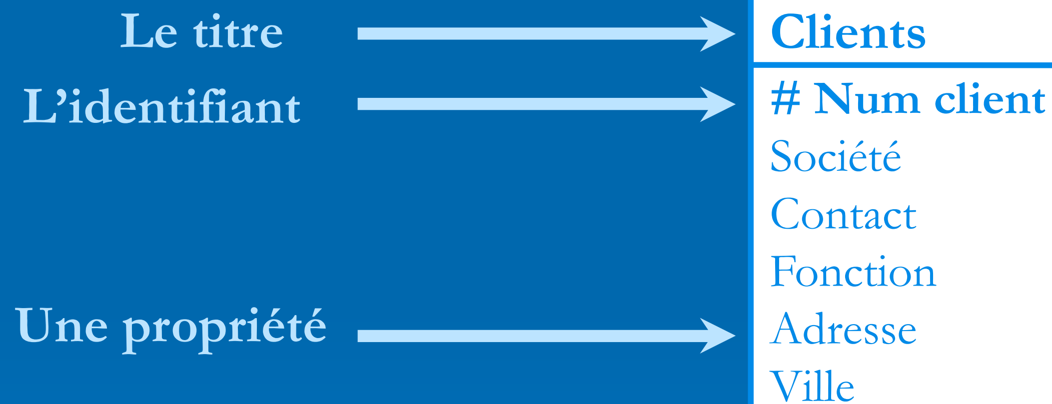
Le concept d'une entité (objet)

Identifiant (clé). Parmi les propriétés d'une entité il peut y avoir une ou plusieurs permettant d'identifier une occurrence unique d'une entité. Si une telle propriété existe, elle identifie l'entité.

Clients
Num client Société Contact Fonction Adresse Ville

Une Occurrence d'entité

Une « valeur »



ENTITE

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

Clients : Table

Code client	Société	Contact	Fonction	Adresse	Ville
ALFKI	Alfreds Futterkiste	Maria Anders	Représentant(e)	Obere Str. 57	Berlin
ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Propriétaire	Avda. de la Constitución 2222	México D.F.
ANTON	Antonio Moreno Taquería	Antonio Moreno	Propriétaire	Mataderos 2312	México D.F.
AROUT	Around the Horn	Thomas Hardy	Représentant(e)	120 Hanover Sq.	London
BERGS	Berglunds snabbköp	Christina Berglund	Acheteur	Berguvsvägen 8	Luleå
BLAUS	Blauer See Delikatessen	Hanna Moos	Représentant(e)	Forsterstr. 57	Mannheim
BLONP	Blondel père et fils	Frédérique Citeaux	Directeur du marketing	24, place Kléber	Strasbourg
BOLID	Bólido Comidas preparadas	Martín Sommer	Propriétaire	C/ Araquil, 67	Madrid
BONAP	Bon app'	Laurence Lebihan	Propriétaire	12, rue des Bouchers	Marseille
BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln	Chef comptable	23 Tsawassen Blvd.	Tsawassen
BSBEV	B's Beverages	Victoria Ashworth	Représentant(e)	Fauntleroy Circus	London
CACTU	Cactus Comidas para llevar	Patricio Simpson	Assistant(e) des ventes	Cerrito 333	Buenos Aires
CENTC	Centro comercial Moctezuma	Francisco Chang	Directeur du marketing	Sierras de Granada 9993	México D.F.
CHOPS	Chop-suey Chinese	Yang Wang	Propriétaire	Hauptstr. 29	Bern

Enr : 1 sur 91

Occurrences d'une propriété

Occurrence d'une entité

Le concept d'une entité (objet)

Notion de dépendance fonctionnelle

Permet de passer d'un ensemble de propriétés non structuré à un modèle conceptuel des données formé d'entités et d'associations.

Pour quoi on dit que P1 est une propriété de E1 et non pas de E2 ?

Définition : une propriété (ou un groupe de propriétés) P2 est dit "fonctionnellement dépendante" d'une propriété (ou un groupe de propriétés) P1 (P1 → P2) si : $a_1 = a_2 \Rightarrow b_1 = b_2$

a_1, a_2, b_1, b_2 étant des valeurs des propriétés P1 et P2 dans une entité du Modèle.

Le concept d'une entité (objet)

Notion de dépendance fonctionnelle

On dit que P1 détermine P2 ou P2 en dépendance fonctionnelle de P1: P1 → P2 si pour chaque valeur de P1 on ne peut faire correspondre qu'une seule valeur de P2. (si $a_1 = a_2$ alors $b_1 = b_2$)

Soient les propriétés suivantes: No_CIN, Nom, Adresse, Age, Profession.

Les dépendances fonctionnelles qui peuvent s'appliquer sont les suivantes :

No_CIN → Nom

No_CIN → Adresse

No_CIN → Age

No_CIN → Profession.

On pourra aussi écrire :

No_CIN → Nom ,Adresse, Age, Profession.

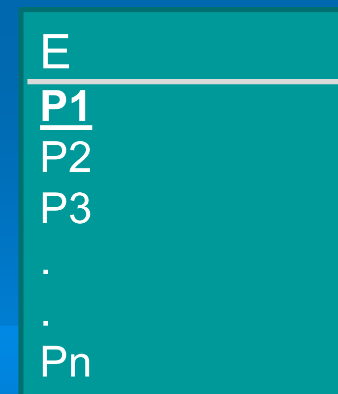
PERSONNEL
No_CIN
Nom
Adresse
Age
Profession

La propriété **No_CIN détermine toutes les autres propriétés**. Il s'agit d'une propriété identifiante d'une entité **Personnel**.

Le concept d'une entité (objet)

Notion de dépendance fonctionnelle

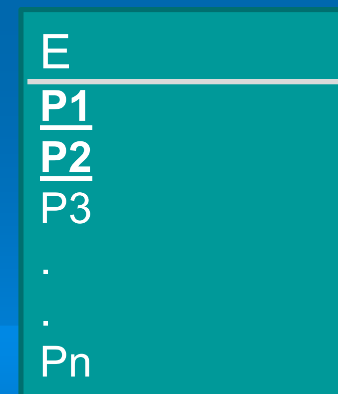
Soit {P1, P2, P3, ……, Pn} un ensemble de propriétés:
Si la propriété P1 détermine toutes les autres propriétés: P1 → P2, P3, Pn. Alors **on peut dire que P1 est un identifiant de d'une entité E** et P2, P3, …Pn sont des propriétés de E



Le concept d'une entité (objet)

Notion de dépendance fonctionnelle composée

Soit {P1, P2, P3, ……, Pn} un ensemble de propriétés:
Si les propriétés P1 et P2 détermine toutes les autres propriétés:
(P1,P2) → P3, P4, …., Pn. Alors **on peut dire que (P1,P2) est un identifiant composé d'une entité E** et P3, P4, …Pn sont des propriétés de E



Le concept d'une entité (objet)

Notion de dépendance fonctionnelle

Soient l'ensemble des propriétés {A, B, C, D, E} dont les valeurs sont données par le tableau suivant:

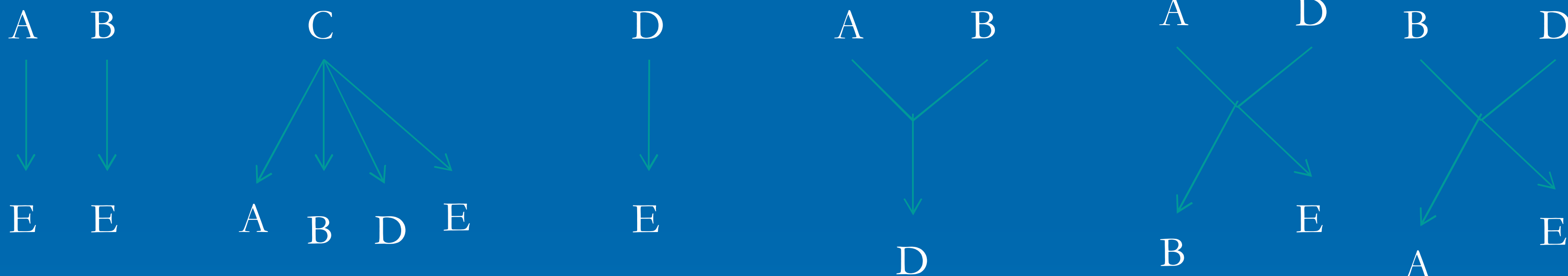
A	B	C	D	E
a1	b1	c1	d1	e1
a1	b2	c2	d2	e1
a2	b1	c3	d3	e1
a2	b1	c4	d3	e1
a3	b2	c5	d1	e1

Quelles sont les DF possibles ?
Existe-il un identifiant ?
Peut-on créer une entité?

Le concept d'une entité (objet)

Notion de dépendance fonctionnelle

Les dépendances fonctionnelles satisfaites par R sont les suivantes :
A -> E ; B -> E ; C -> ABDE ; D -> E ; AB -> D ; AD -> BE ; BD -> AE.



Entité
C A B D E

Le concept d'une entité (objet)

Notion de dépendance fonctionnelle directe

Pour supprimer les redondances, nous devons avoir **une dépendance fonctionnelle directe** entre l'identifiant de l'entité et les autres propriétés non identifiantes de l'entité.

Une dépendance fonctionnelle x → y est directe s'il n'existe pas de propriété z telle que : x → z et z → y.

Article
Référence
Désignation
PUHT
N°Catégorie
LibelléCatégorie

AE23	AZ27	BE33
Imprimante	Disque-Dur	Toner
3000,00	1000,00	1000,00
A	A	B
Mat-Info	Mat-Info	Conso-info

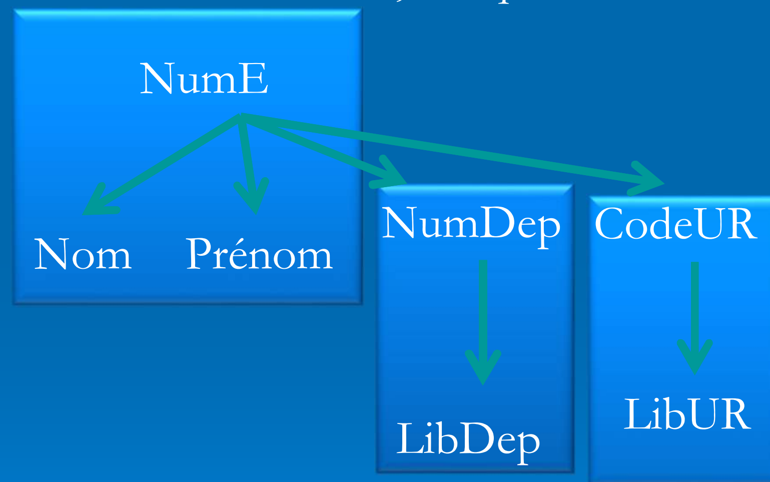
Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

DFD Exemple-1

Soit les propriétés suivantes:
{NumE, Nom, Prénom, NumDep, LibDep, CodeUR, LibelléUR} tel que:

NumE → Nom
NumE → Prénom
NumE → NumDep
NumE → CodeUR
NumDep → LibDep
CodeUR → LibUR



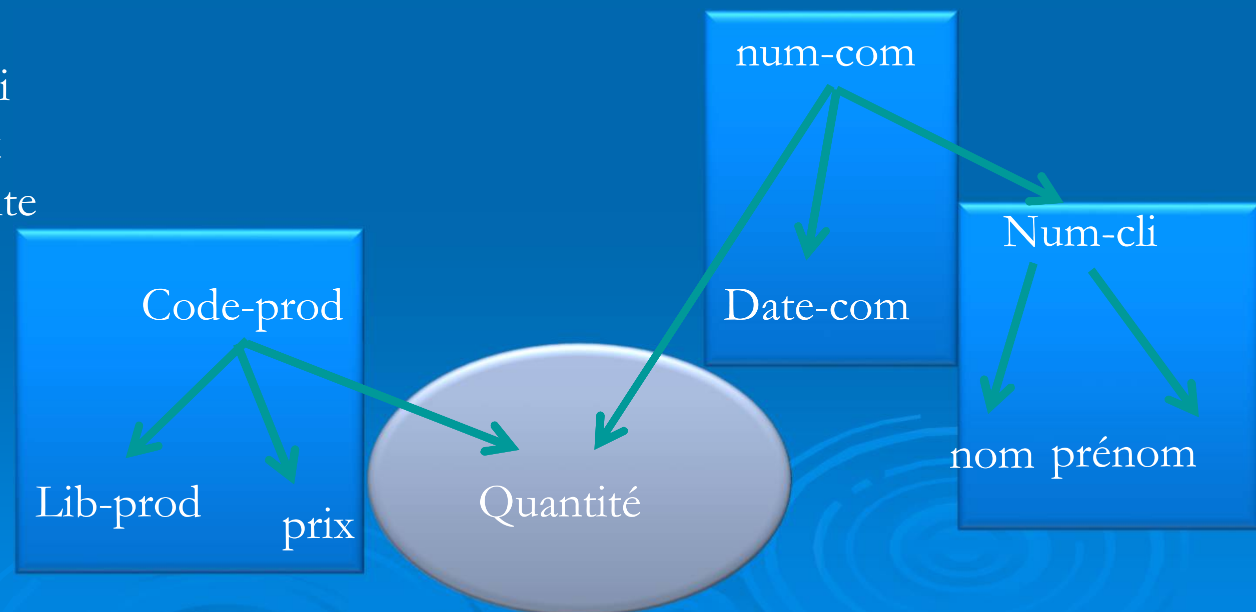
Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

DFD Exemple-2

Soit les propriétés suivantes :
num-cli, nom, adresse, num-com, date-com, code-prod, prix, lib-prod, quantite:

num-cli --> nom, adresse
num-com --> date-com, num-cli
code-prod --> lib-prod, prix
code-prod, num-com --> quantite



Modèle Conceptuel de Données (MCD)

Le concept d'une association (Relation)

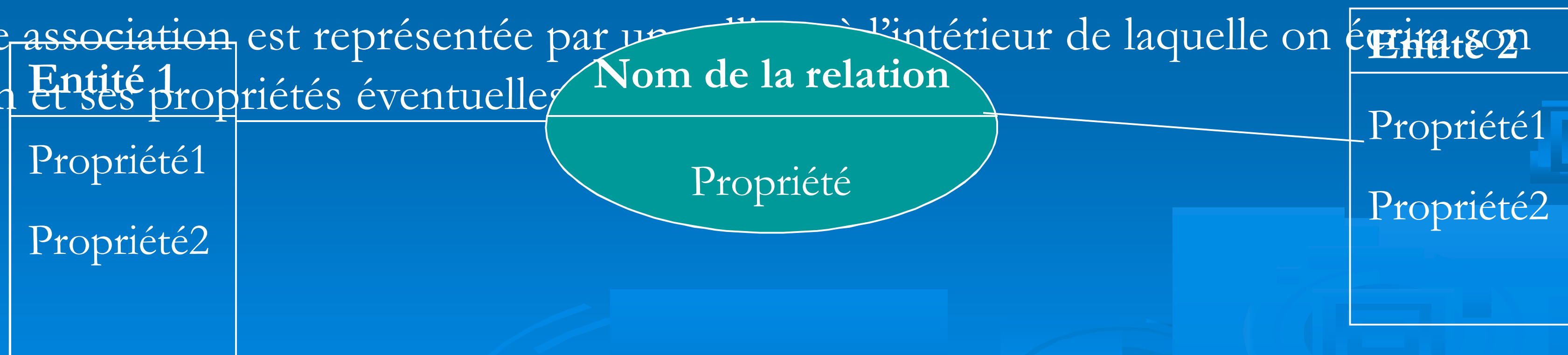
Définition

Une association traduit les rapports ou les relations existants entre deux ou plusieurs entités.

Occurrence d'association. une correspondance entre deux ou plusieurs occurrences d'entités afin de conserver les informations

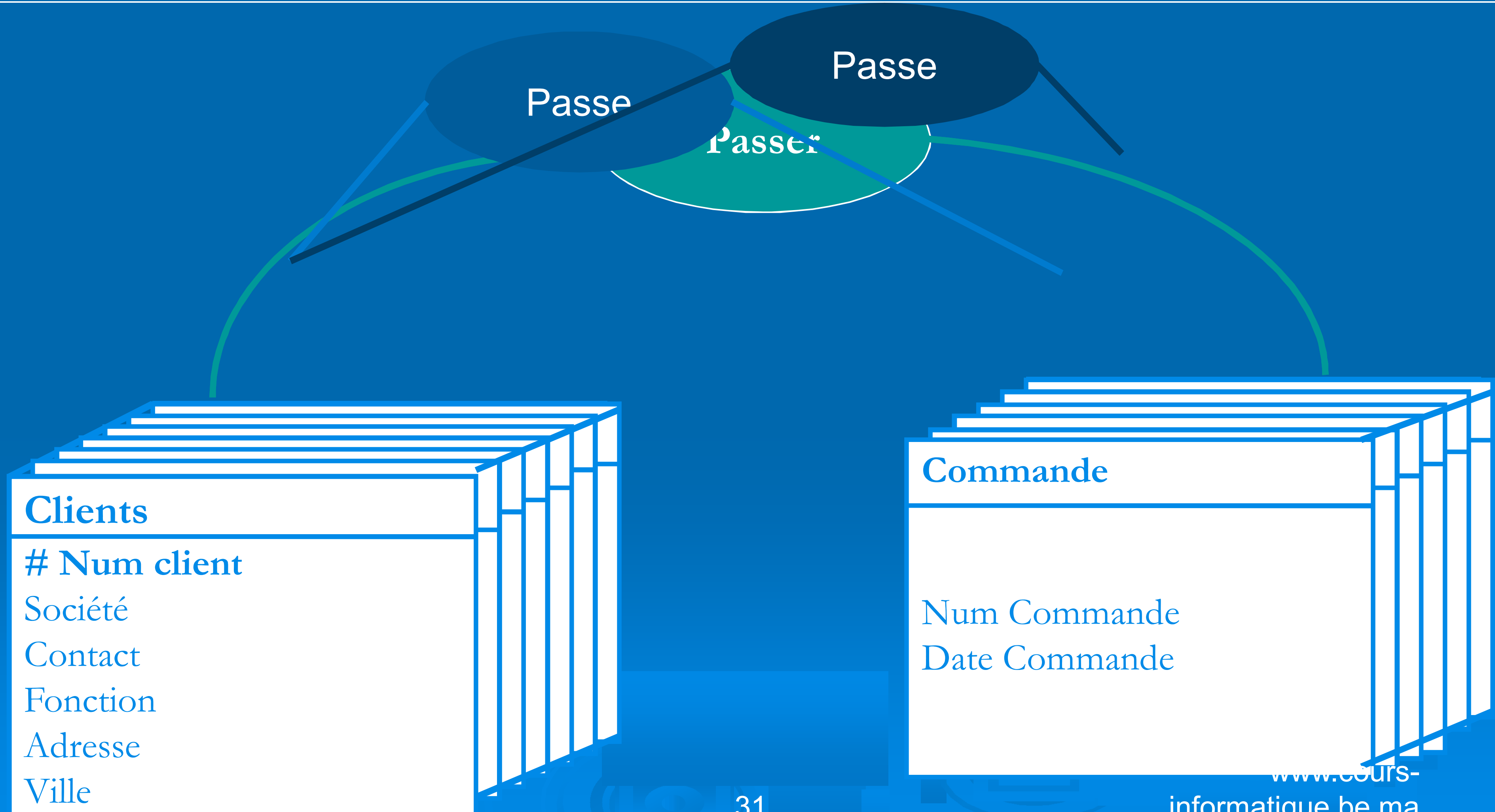
Formalisme

Une association est représentée par une ellipse à l'intérieur de laquelle on écrira son nom et ses propriétés éventuelles



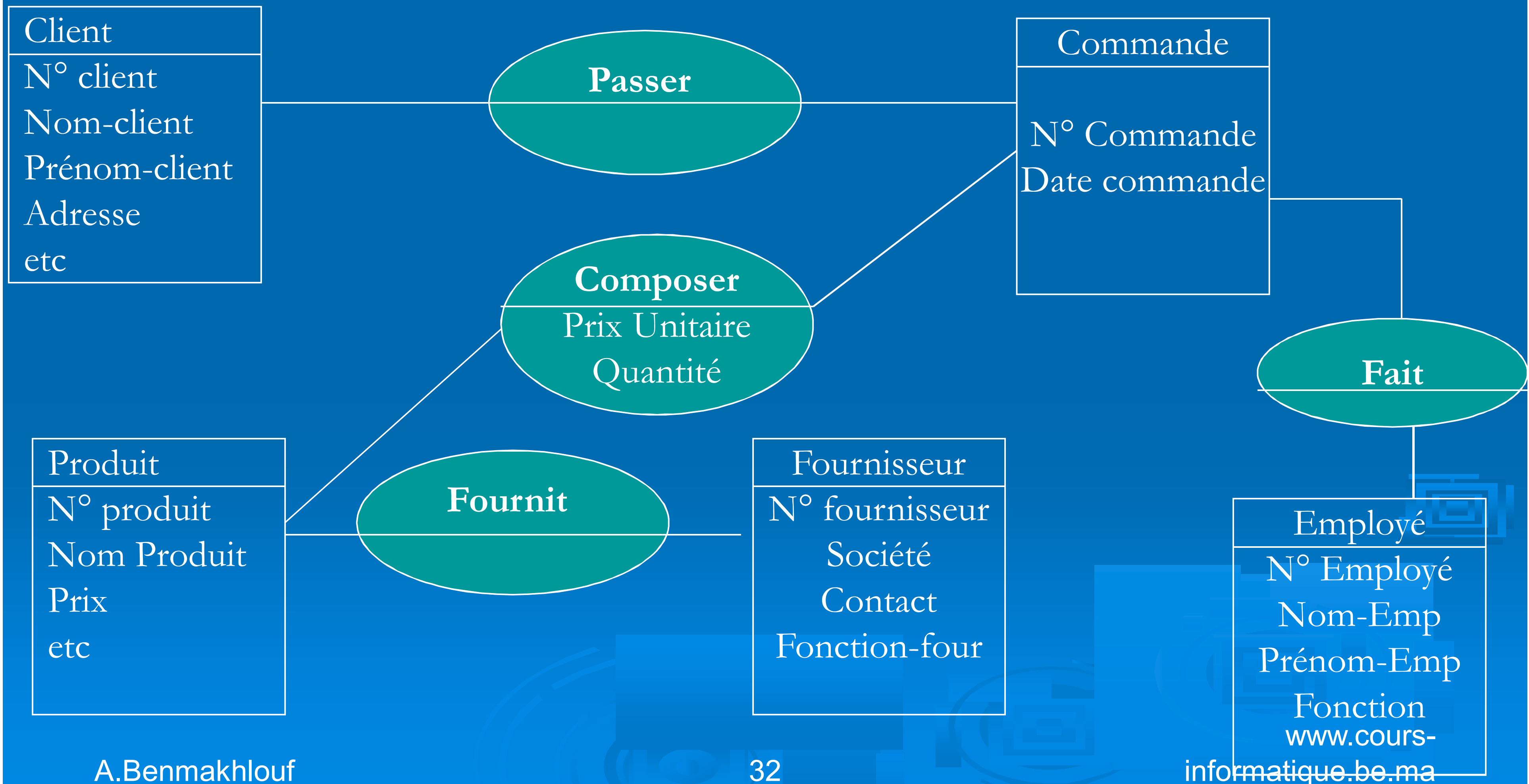
Modèle Conceptuel de Données (MCD)

Le concept d'une association (Relation)



Modèle Conceptuel de Données (MCD)

Le concept d'une association (Relation)



Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)

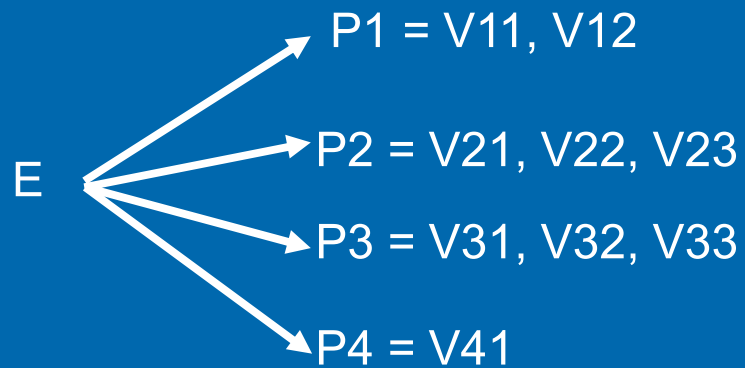
Entité : Objets qui peut être décrit avec des propriétés appelées attributs

Une Propriété (attribut) : c'est une caractéristique ou une qualité d'une entité ou d'une association". Il peut prendre une (ou plusieurs) valeur(s).

"**Une valeur** est un symbole utilisé pour représenter un fait élémentaire".

Modèle Conceptuel de Données (MCD)

Le concept d'une entité (objet)



Le concept d'une association (Relation)

La cardinalité d'une relation

C'est une étape essentielle de la démarche de conception de base de donnée.

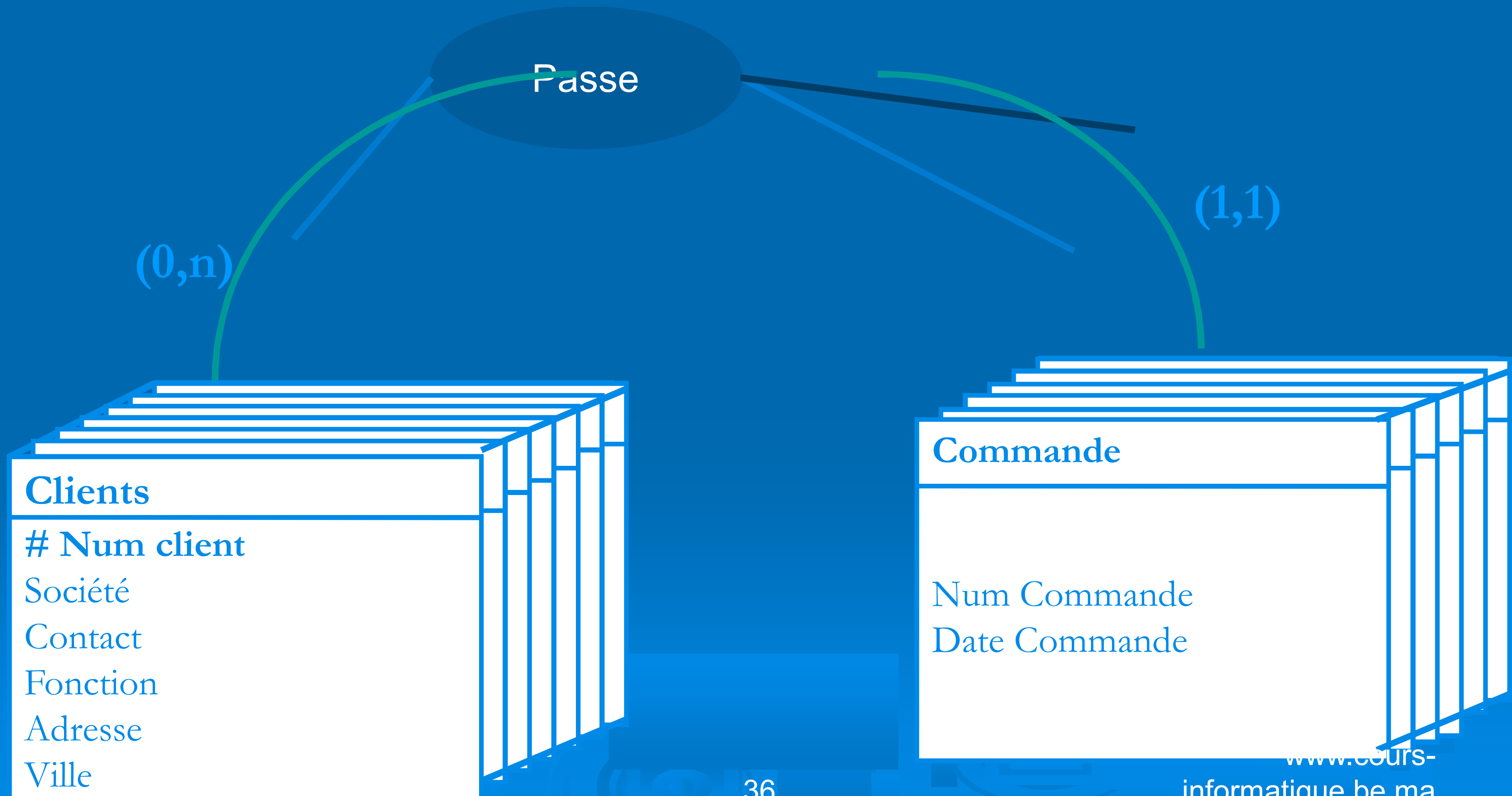
C'est le nombre de fois minimum et maximum de participation d'une occurrence d'une entité à une relation.

La cardinalité s'exprime par deux nombres appelés cardinalité minimale et cardinalité maximale

- La cardinalité minimale prend les valeurs 0 ou 1
- La cardinalité maximale prend les valeurs 1 ou n

Modèle Conceptuel de Données (MCD)

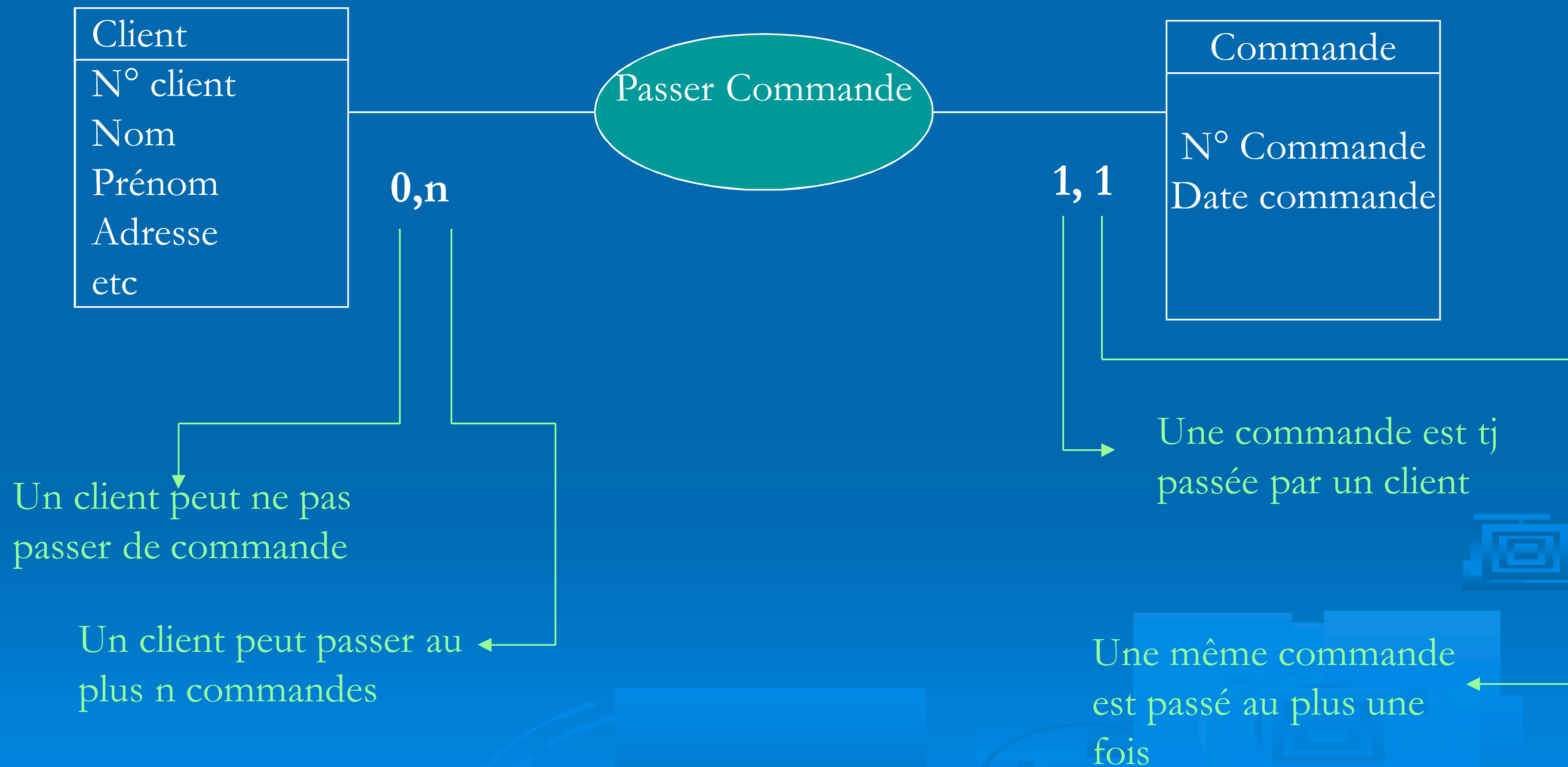
Le concept d'une association (Relation)



www.cours-informatique.be.ma

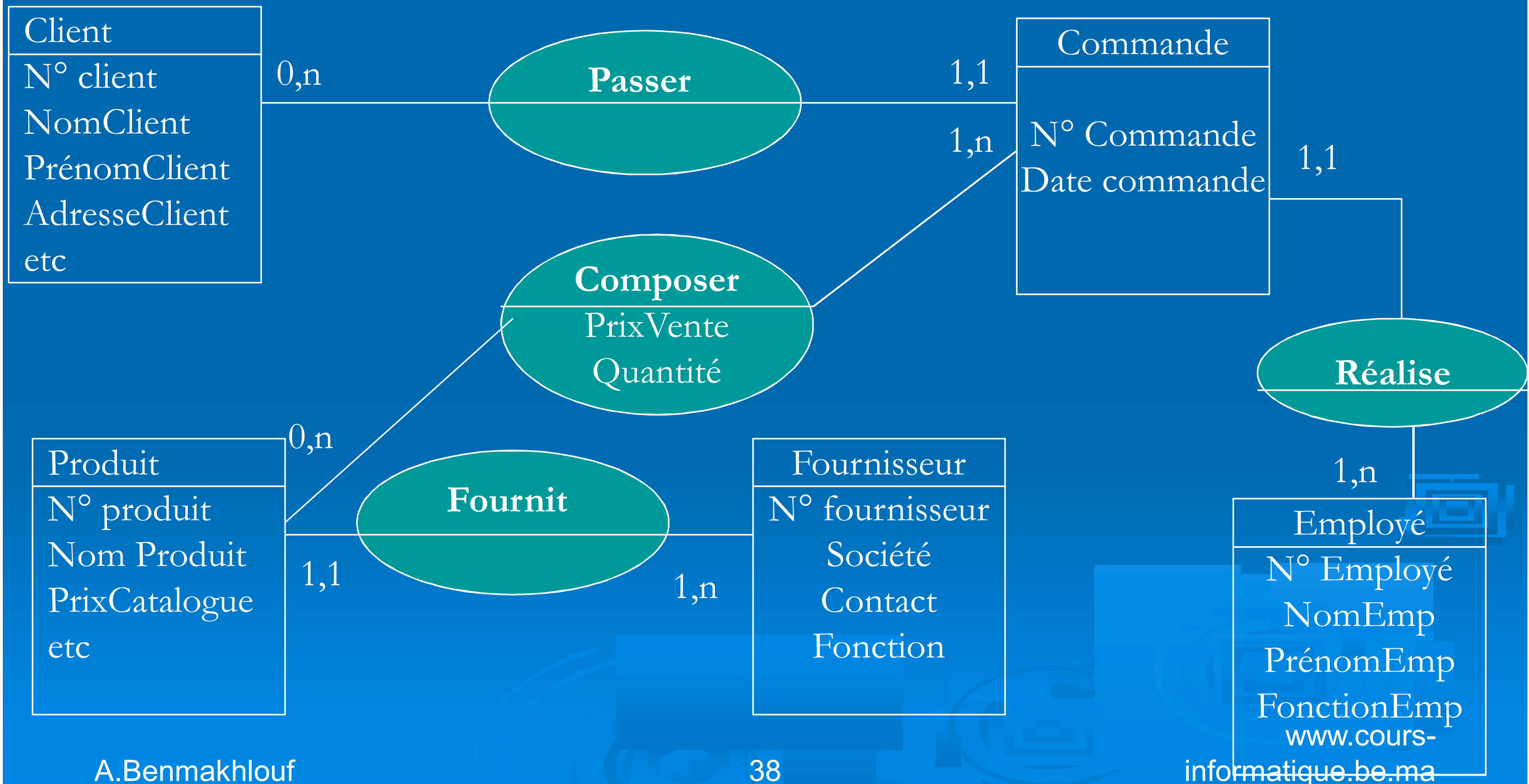
Modèle Conceptuel de Données (MCD)

Le concept d'une association (Relation)



Modèle Conceptuel de Données (MCD)

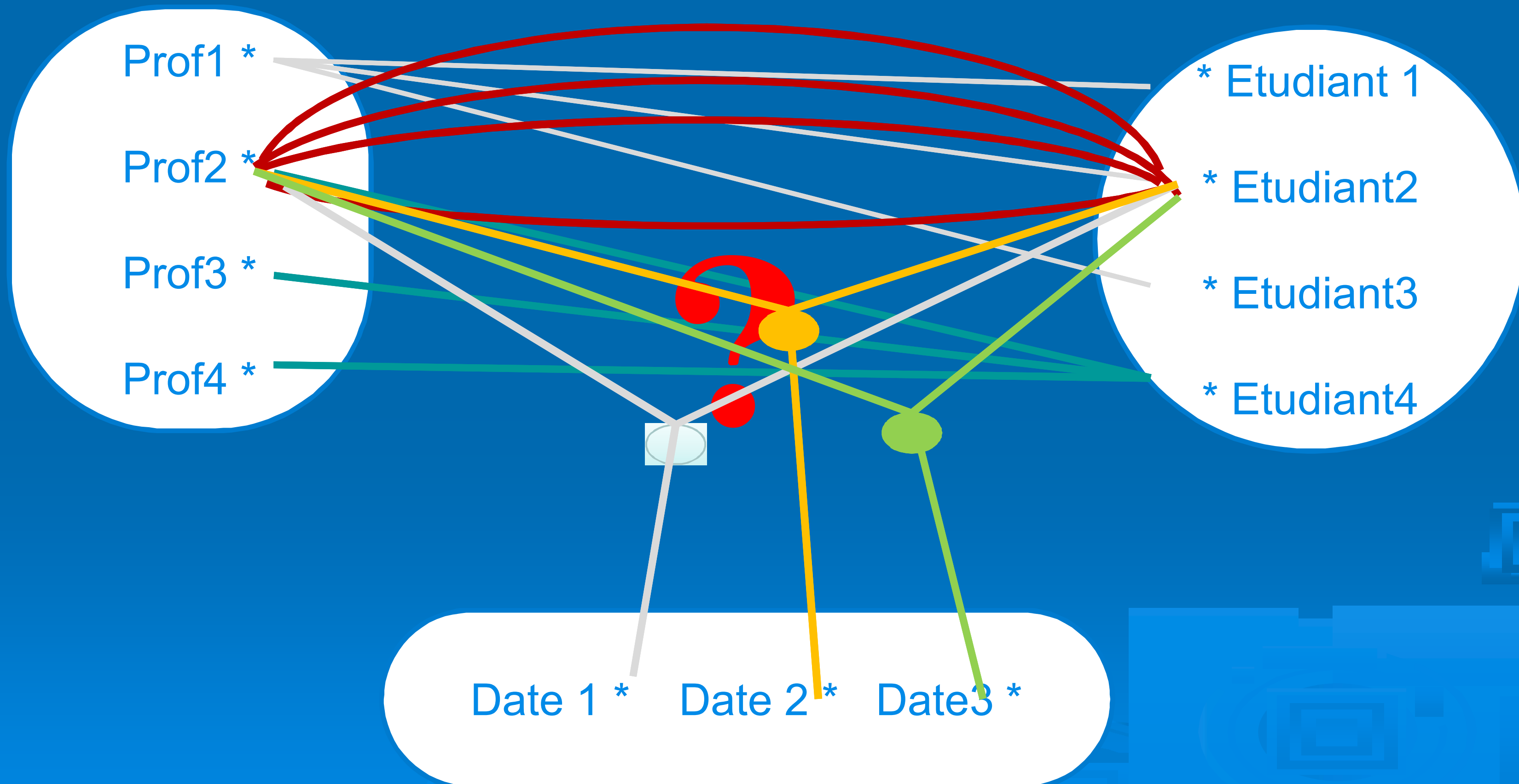
Le concept d'une association (Relation)



Modèle Conceptuel de Données (MCD)

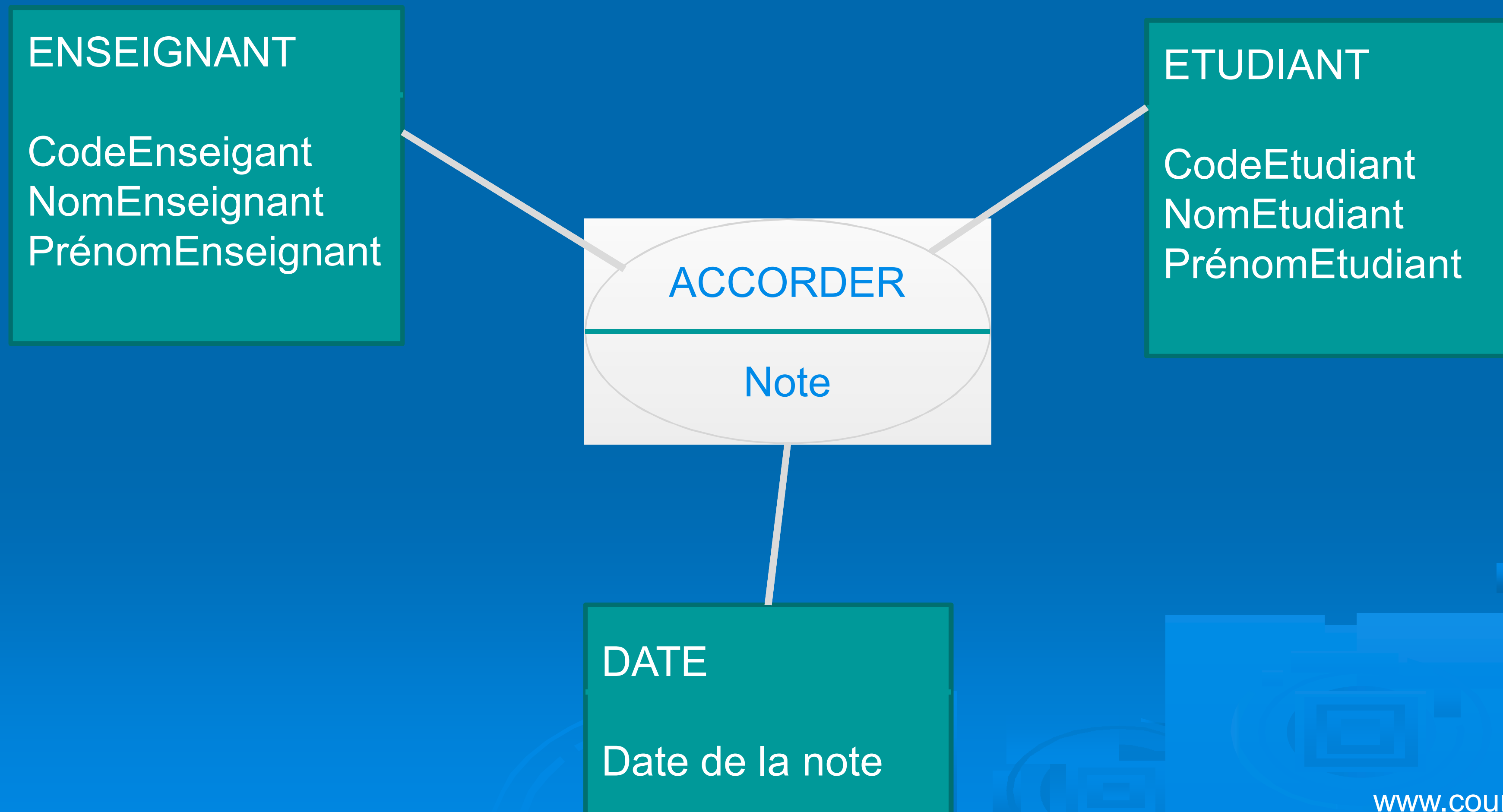
Le concept d'une association (Relation)

Les profs accordent des notes aux étudiants



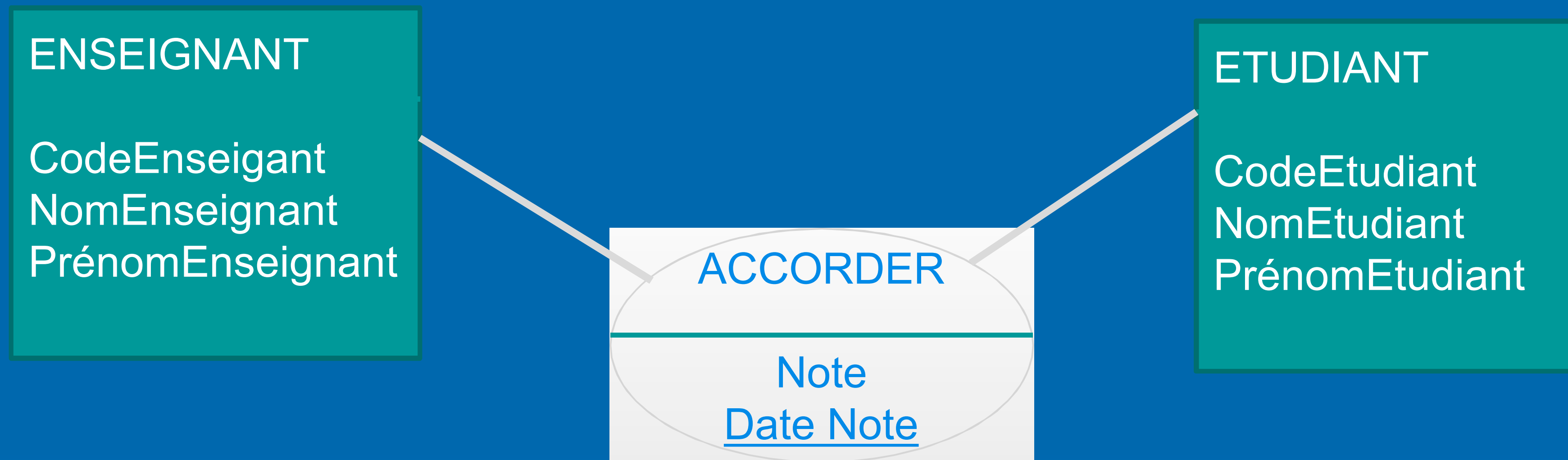
Modèle Conceptuel de Données (MCD)

Le concept d’une association (Relation)



Modèle Conceptuel de Données (MCD)

Le concept d'une association (Relation)



Modèle Conceptuel de Données (MCD)

Exemple récapitulatif d'élaboration d'un MCD

Soit les données suivantes

Date de commande
Nom client
Désignation produit
Adresse client
N° client
Référence produit
Prix unitaire de vente
Quantité commandé
N° commande
Montant HT/produit
Prix unitaire catalogue
Délai de livraison
Montant total HT
Nombre de ventes/produit

1- quelles sont les données qui ne peuvent pas être des propriétés

2- élaborer un MCD en se basant sur la DFD

Modèle Conceptuel de Données (MCD)

Exemple récapitulatif d'élaboration d'un MCD

Soit une PME spécialisée dans la mise à disposition des employés pour le compte de ses **clients**.

✓Chaque intervention donne lieu à un **contrat** avec le **client**: les principales informations du contrat sont:

- N° du contrat
- La description de l'intervention
- Date début de l'intervention

✓ Ce contrat définie les qualifications de chaque intervenant (il existe une vingtaine de qualifications possibles) et le nombre de jour d'intervention correspondant à chaque qualification

A chaque **qualification** correspond un tarif journalier. La PME s'accorde en interne une certaine souplesse sur la détermination précise de la qualification de son personnel en procédant de la manière suivante:

✓ Chaque **employé** possède a priori une qualification de base.

✓ A chaque intervention il est possible de réajuster la qualification. Donc il faut connaitre le degrés de compétence de chaque employé dans les autres qualifications.

www.cours-informatique.be.ma

Modèle Conceptuel de Données (MCD)

Exemple récapitulatif d'élaboration d'un MCD

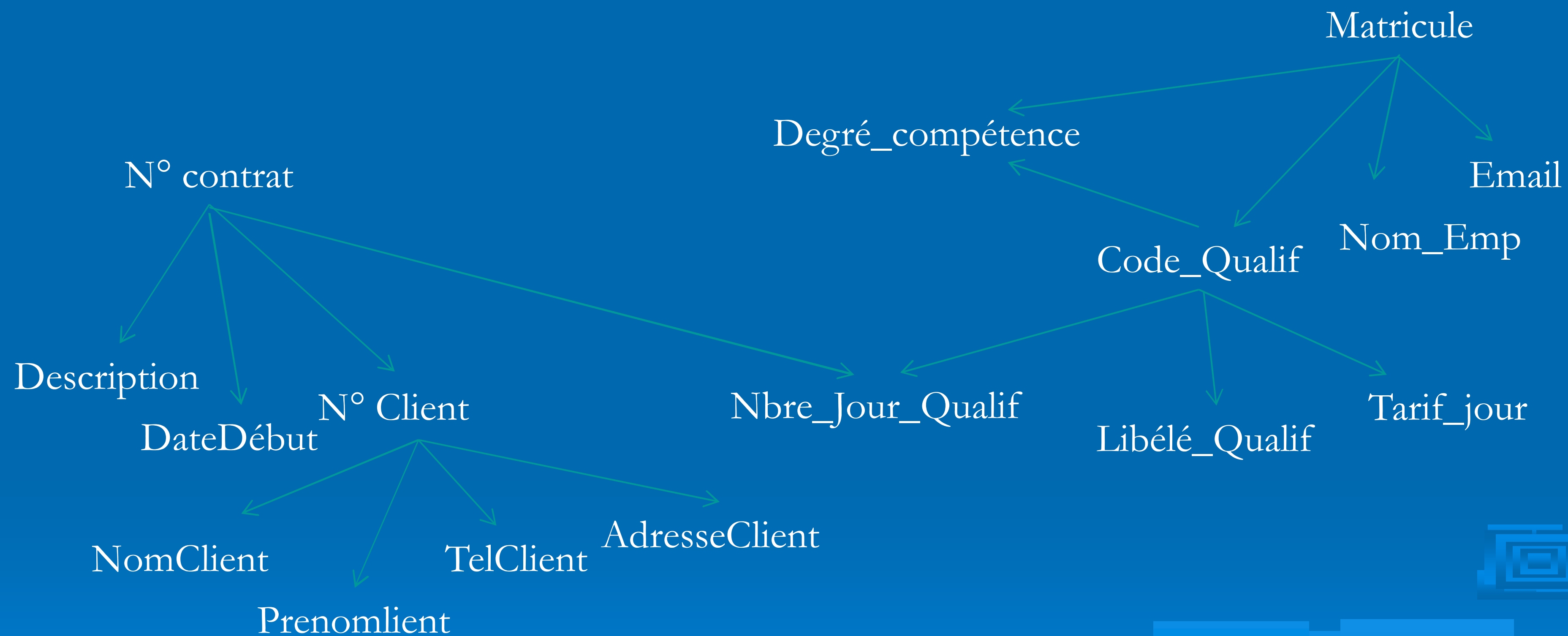
1- **Dictionnaire des données** (les entités, les noms de différentes propriétés, les associations).

Num Client
Nom client
Adresse
Raison sociale
Ville
Num Contrat
Objet
Date début

Num Employé
Nom
Prénom
Etc
Code qualif
Libelé qualif
Tarif/jour
Nbre_Jour/Qualif
Degré_compétence

Modèle Conceptuel de Données (MCD)

Diagramme de dépendance



Modèle Conceptuel de Données (MCD)

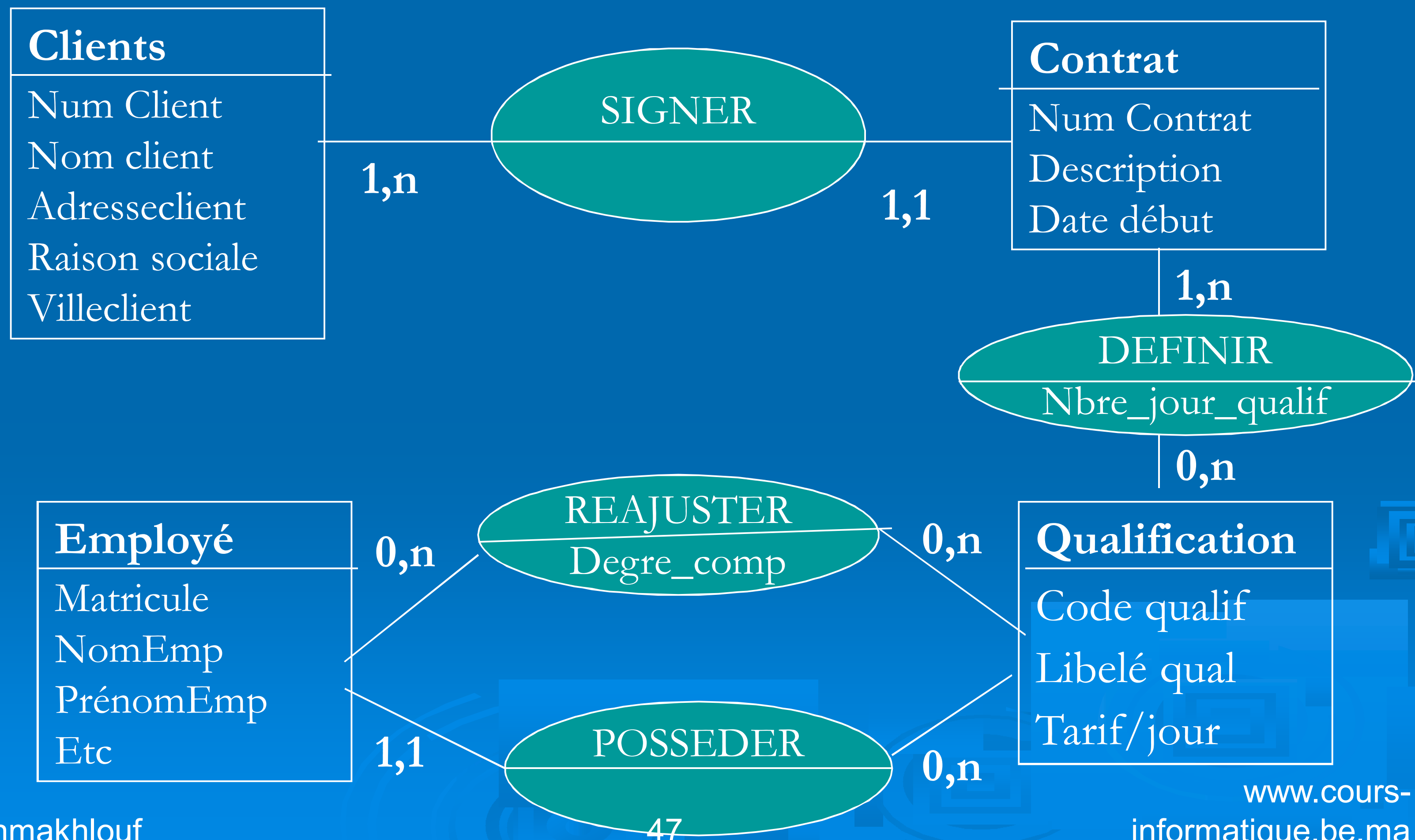
Exemple récapitulatif d'élaboration d'un MCD

2- Calcule des cardinalités

Entité	Association	Card.	Justification
Client	Passe	1,n	Un client peut passer 1 contrat comme il peut passer n contrats
Contrat	Est passé	1,1	Un contrat est passé par un client au plus
Contrat	Définit	1,n	Un Contrat définit au moins une qualification
Qualification	Est définit	0 ,n	Une qualification peut na pas être définie comme elle peut être définie par n contrats
Employé	Réajuster	0,n	Un Employé peut ne réajuster aucune qualification comme il peut réajuster n qualifs
Qualification	Être réajuster	0,n	Une qualifs peut ne pas être réajuster par aucun employé comme elle peut être réajuster par n employés
Employé	Posséder	1,1	Chaque employé possède une qualification
Qualification	Est possédé	0,n	Une qualif peut ne pas être possédé par aucun employé comme elle peut être possédé par n employés

Modèle Conceptuel de Données (MCD)

Exemple récapitulatif d'élaboration d'un MCD



Modèle Logique de Données Relationnel (MLDR)

C'est l'étape qui consiste à transposer le MCD en Modèle Logique de Données Relationnelles (MLDR).

Ce MLDR est en fait le dernier pas vers le Modèle Physique de donnée (MPD)



Des règles strictes, nécessaires et suffisantes pour passer d'un MCD à un MLDR

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Relationnel

C'est un modèle qui s'inspire du concept mathématique de **relation**.

Exemple

Soients les domaines **A = {1, 2, 3}** et **B = {a, b}.**

Le **produit cartésien** de A et B est donné par:

A x B = {(1, a), (2, a), (3, a), (1, b), (2, b), (3, b)}.

Le sous - ensemble $\Re$= {(1, a), (1, b), (2, a), (2,b)} est une relation définie sur les **domaines** A et B.

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Relationnel

La relation ℜ peut s'écrire, également, sous la forme :

ℜ	SA	SB
	1	a
	1	b
	2	a
	2	b

Avec SA et SB sont des sous ensemble des domaines A et B

la relation ℜ est représentée comme un tableau de données appelé Table ou Tableau

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Relationnel

Exemple.

Considérons les domaines **Société = {Nom1, Nom2, Nom3, Nom4, Nom5, ….}** et **Ville**={Casablanca, Rabat, Fès, Marrakech, ….}. Considérons la relation

Client = {(Nom1, Casablanca), (Nom2, Rabat), (Nom3, Rabat), (Nom4, Casablanca)}.

Attribut ou Champ ou Rubrique

Client	NomClient	VilleClient
	Nom1	Casablanca
	Nom2	Rabat
	Nom3	Rabat
	Nom4	Casablanca

Enregistrement

Ou

Occurrence

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Relationnel

Une relation est représentée par son nom suivi de la liste de ses champs. Par exemple la relation Voiture s'écrit : **Client(**NomClient, VilleClient**).**

Clé. On entend par **clé** un ou plusieurs champs permettant d'identifier un enregistrement unique de la relation.

Client	CodeClient	NomClient	VilleClient
	CL1	Nom1	Casablanca
	CL2	Nom2	Rabat
	CL3	Nom3	Rabat
	CL4	Nom4	Casablanca

On écrira la Relation:

Client(<u>CodeClient</u>, NomClient, VilleClient**).**

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

Règles de passage : Entités

- Toute **entité** du MCD devient une **relation** du MR, et donc une table de la Base de Donnée. La relation porte le même nom que l'entité

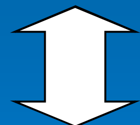
- Chaque **propriété de l'entité** devient un **champ de la relation**,

- **L'identifiant** de l'entité devient la **Clé Primaire** de la relation

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

Clients
<u>Num client</u>
Société
Contact
Fonction
Adresse
Ville



Client (<u>Num client</u>, Société, Contact, Fonction, Adresse, Ville)

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

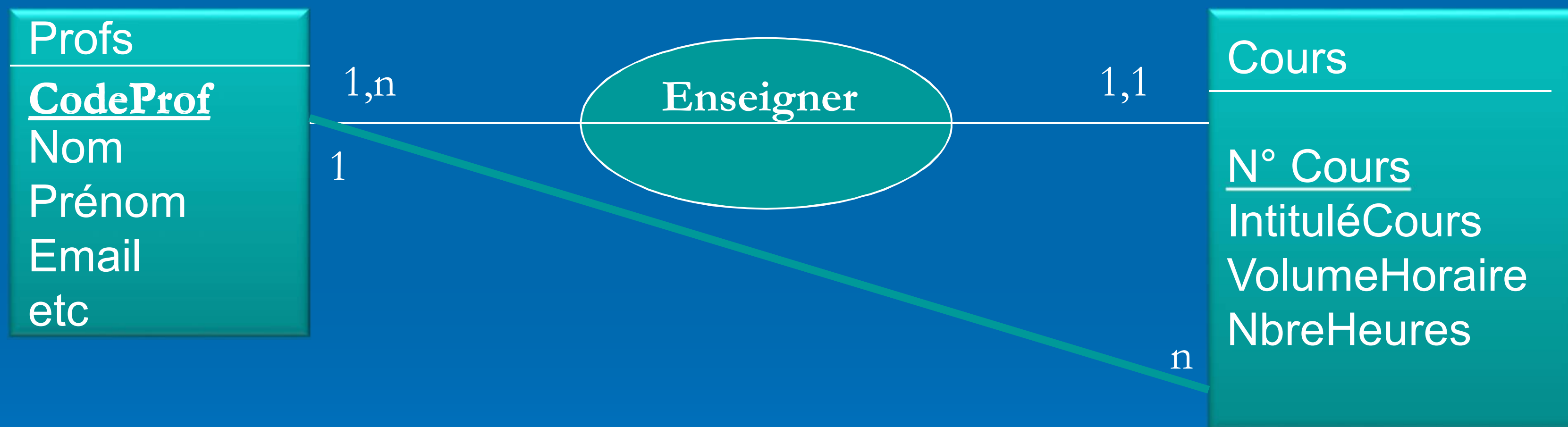
Régles de passage : Association binaire aux cardinalités (X,1) - (X,n), X=0 ou X=1

La **Clé Primaire** de la table à la cardinalité (X,n) devient un attribut et donc une **Clé Etrangère** dans la table à la cardinalité (X,1) :

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

Règles de passage : Association binaire aux cardinalités (X,1) - (X,n), X=0 ou X=1



Prof(CodeProf, Nom, Prénom, Email)

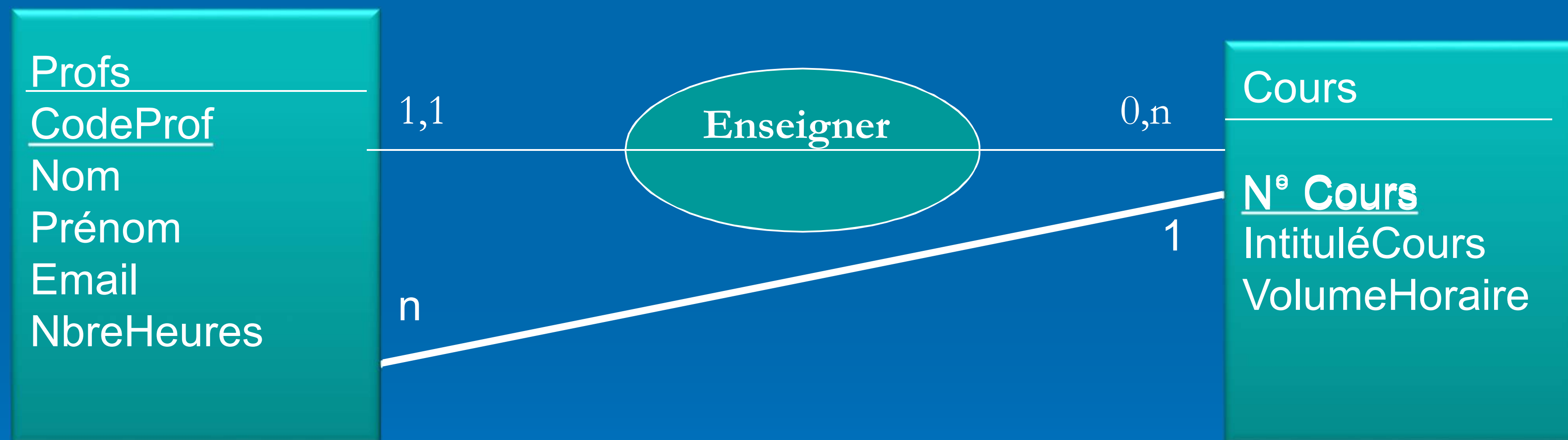
Cours(N°Cours, Intitulé, VolumeHoraire, NbreHeures, #CodeProf)

NbreHeures/Prof ????

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

Règles de passage : Association binaire aux cardinalités (X,1) - (X,n), X=0 ou X=1



Prof(CodeProf, Nom, Prénom, Email, NbreHeures, #N°Cours)
Cours(N°Cours, Intitulé, VolumeHoraire)

NbreHeures/Cours ????

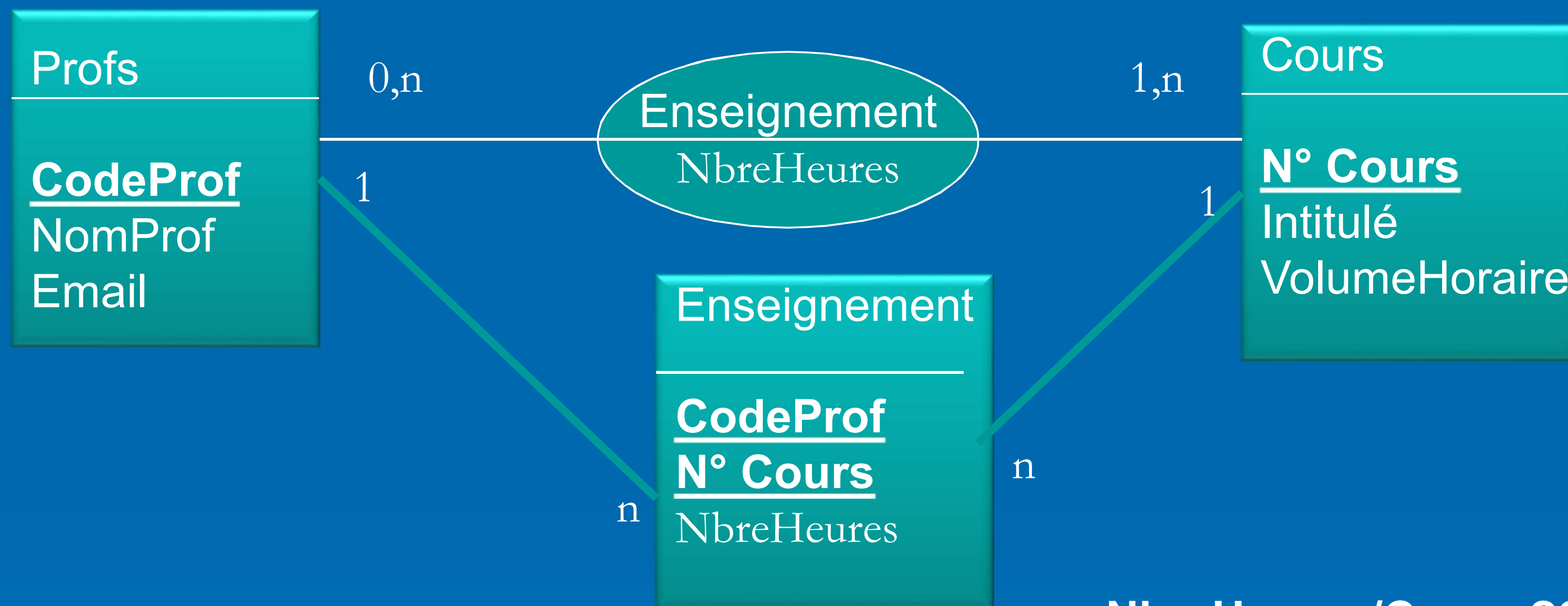
Modèle Logique de donnée Relationnel (MLDR)

Règles de passage : Association binaire aux cardinalités (X,n) - (X,n), X=0 ou X=1

L'association se transforme en Relation (table) ayant comme **Clé Primaire la concaténation (**une clé composée) des **identifiants** des 2 entités.

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)



NbreHeures/Cours ????

NbreHeures/Prof ????

Profs (<u>CodeProfs</u>, NomProfs, Email)

Cours (<u>N°Cours</u>, intitulé, VolumeHoraire)

Enseignement(<u>CodeProfs, N°Cours</u>, Nbre heures)

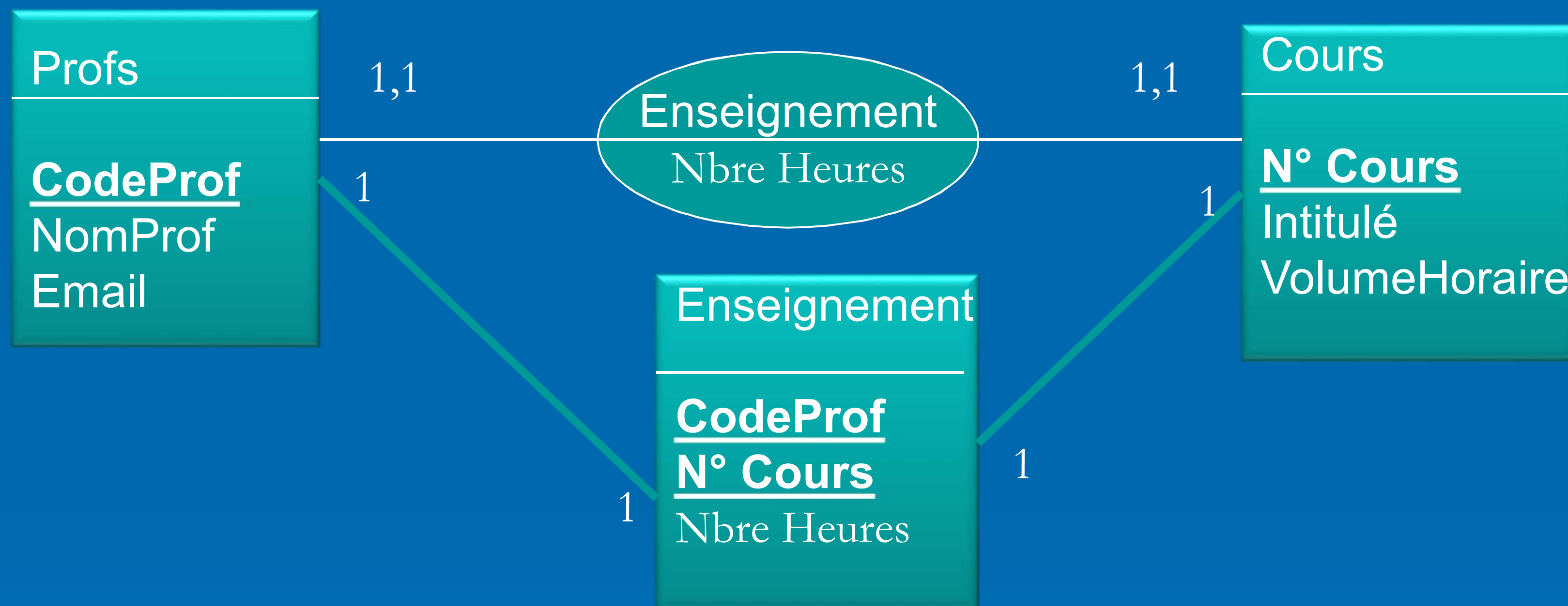
Modèle Logique de donnée Relationnel (MLDR)

Règles de passage : Association binaire aux cardinalités (1,1) - (1,1)

L'association se transforme en Relation (table) ayant comme **Clé Primaire la concaténation (**une clé composée) des **identifiants** des 2 entités.

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)



Profs (<u>CodeProfs</u>, NomProfs, Email)
Cours (<u>N°Cours</u>, intitulé)
Enseignement(CodeProfs, N°Cours, Nbre heures)

Modèle Logique de donnée Relationnel (MLDR)

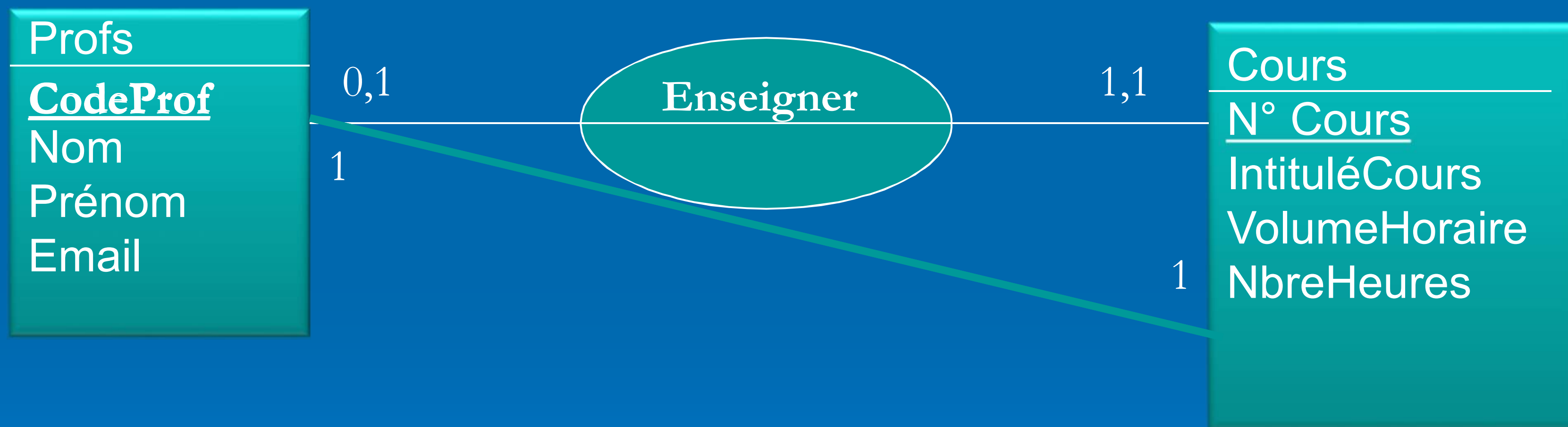
Régles de passage : Association binaire aux cardinalités (0,1) - (1,1)

La **Clé Primaire** de la table à la cardinalité (0,1) devient un attribut et donc une **Clé Etrangère** dans la table à la cardinalité (1,1) :

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

Règles de passage : Association binaire aux cardinalités (X,1) - (X,1), X=0 ou X=1

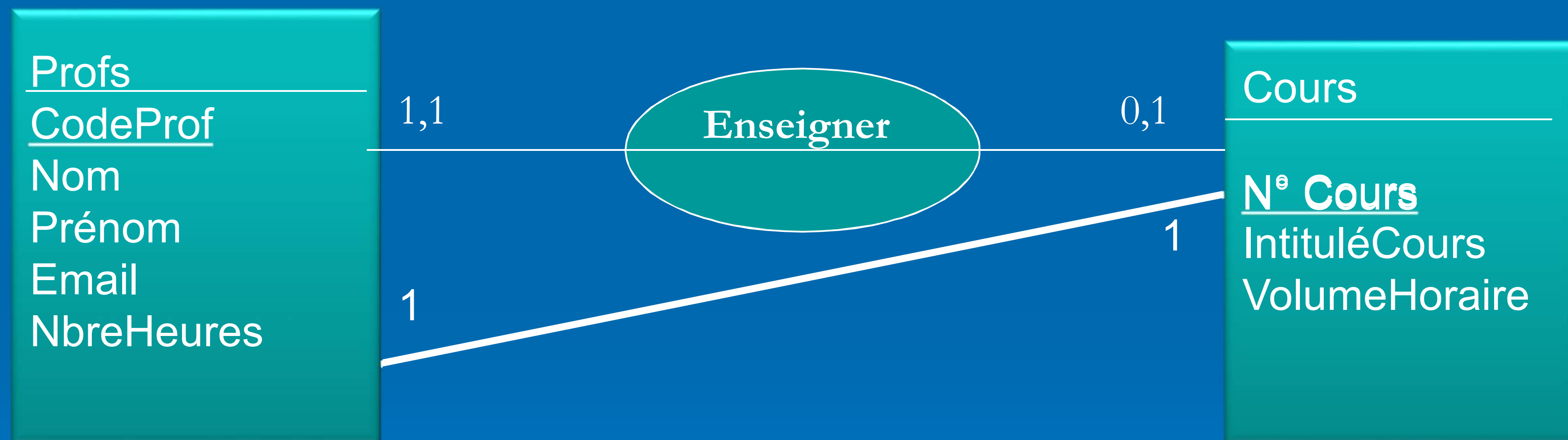


Profs(CodeProf, Nom, Prenom, Email)
Cours (N°Cours, Intitulé, VolumeHoraire, NbreHeures, CodeProf)

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

Règles de passage : Association binaire aux cardinalités (X,1) - (X,1), X=0 ou X=1

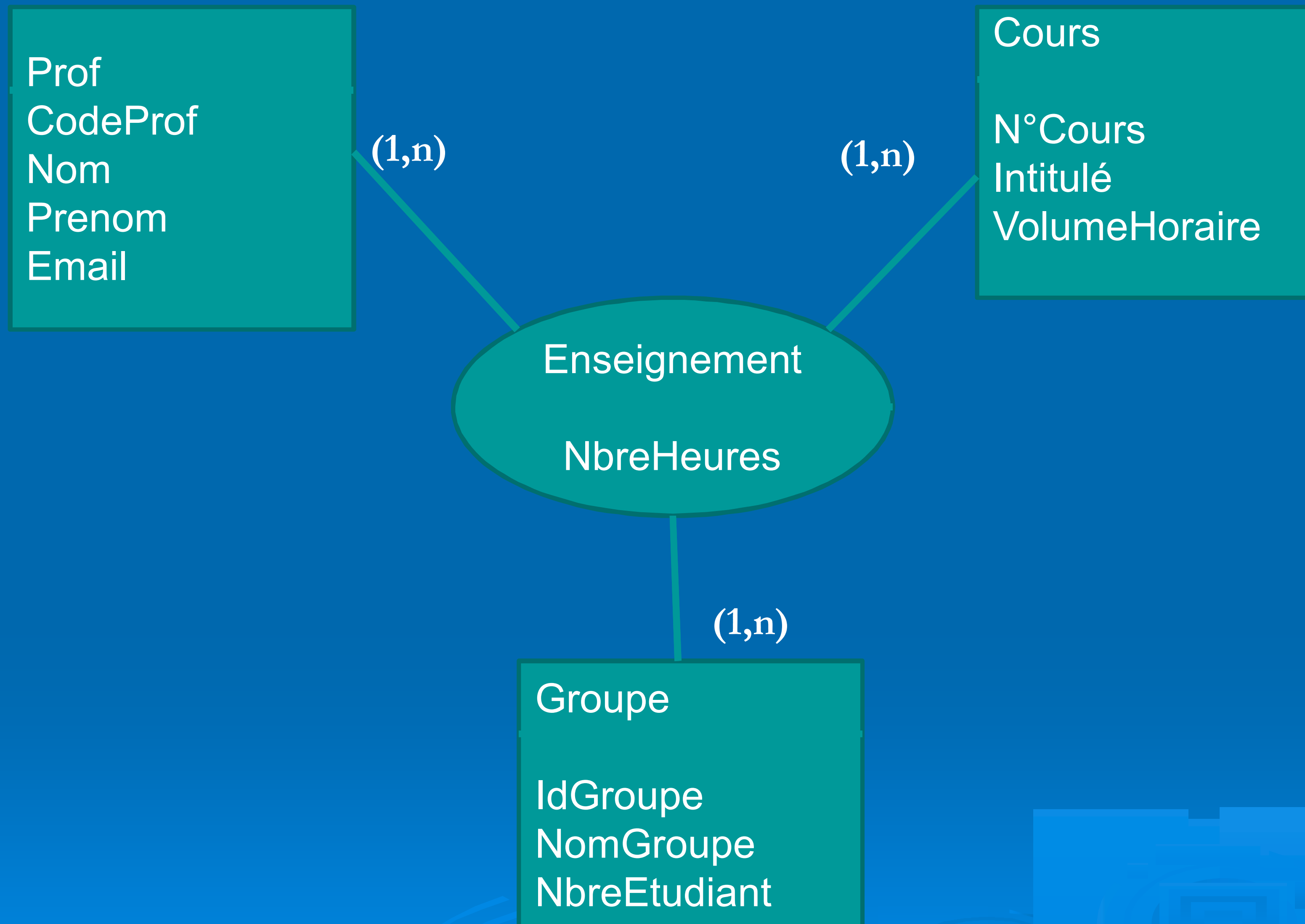


Prof(CodeProf, Nom, Prénom, Email, NbreHeures #N°Cours)
Cours(N°Cours, Intitulé, VolumeHoraire)

Modèle Logique de donnée Relationnel (MLDR)

Relation n-aire (quelle que soit les cardinalités)

Il y a création d'une table supplémentaire ayant comme **Clé Primaire** la **concaténation** des **identifiants** des entités participant à la relation.


Prof
CodeProf
Nom
Prenom
Email
(1,n)
(1,n)
Cours
N°Cours
Intitulé
VolumeHoraire
Enseignement
NbreHeures
(1,n)
Groupe
IdGroupe
NomGroupe
NbreEtudiant

Modèle Logique de Donnée Relationnel (MLDR)

Modèle Logique de donnée Relationnel (MLDR)

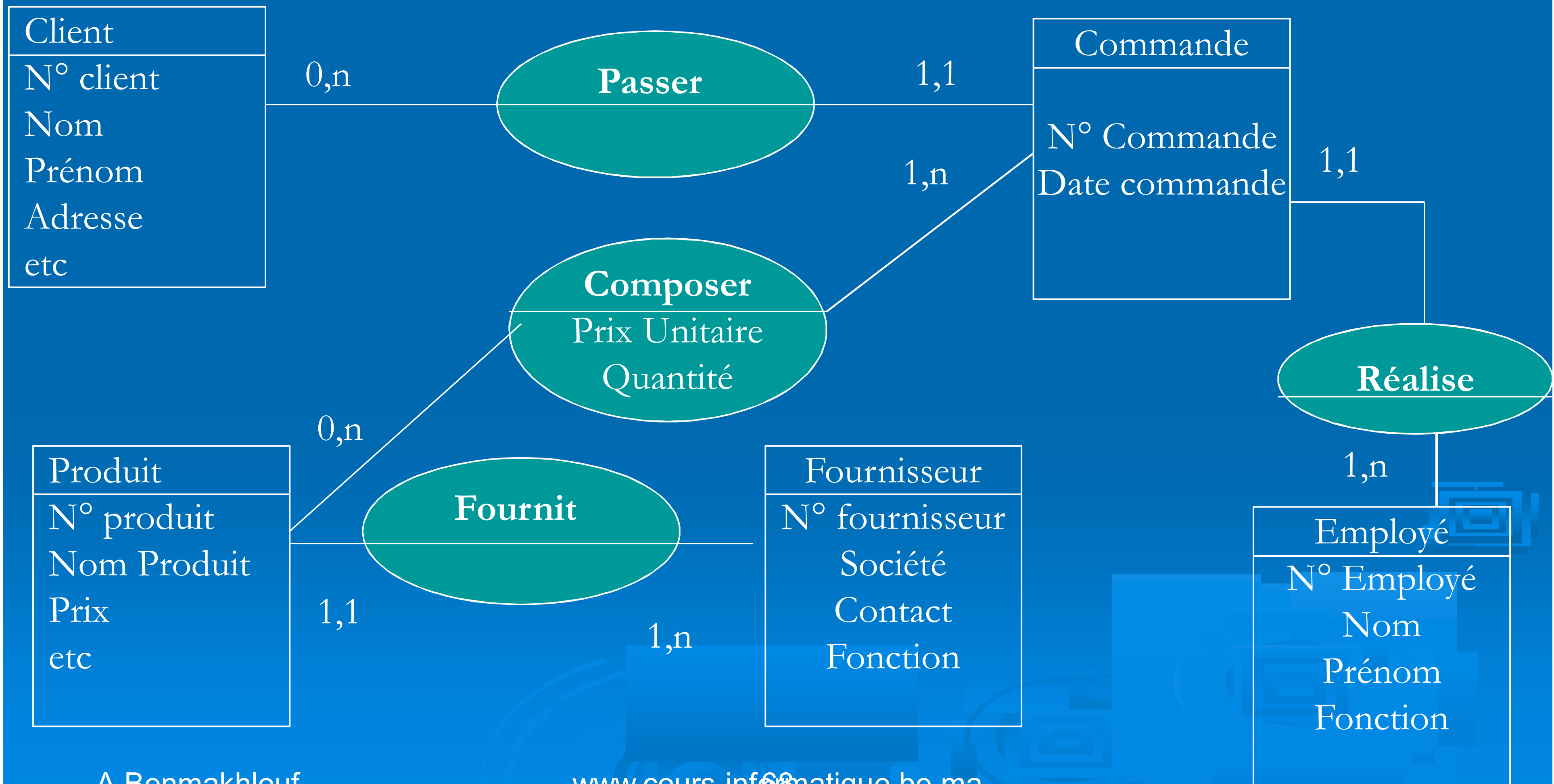
Relation n-aire (quelle que soit les cardinalités)

PROF(CodeProf, Nom, Prénom, Email)
COURS(N°Cours, Intitulé, VolumeHoraire)

GROUPE(**IdGroupe**, NomGroupe, NbreEtudiant)
Enseignement(CodeProf, N°Cours, IdGroupe, NbreHeures)

Modèle Conceptuel de Donnée (MCD)

Exemple



Modèle Logique de Donnée (MLDR)

Client (<u>N° Client</u>, nom, prénom, Adresse)

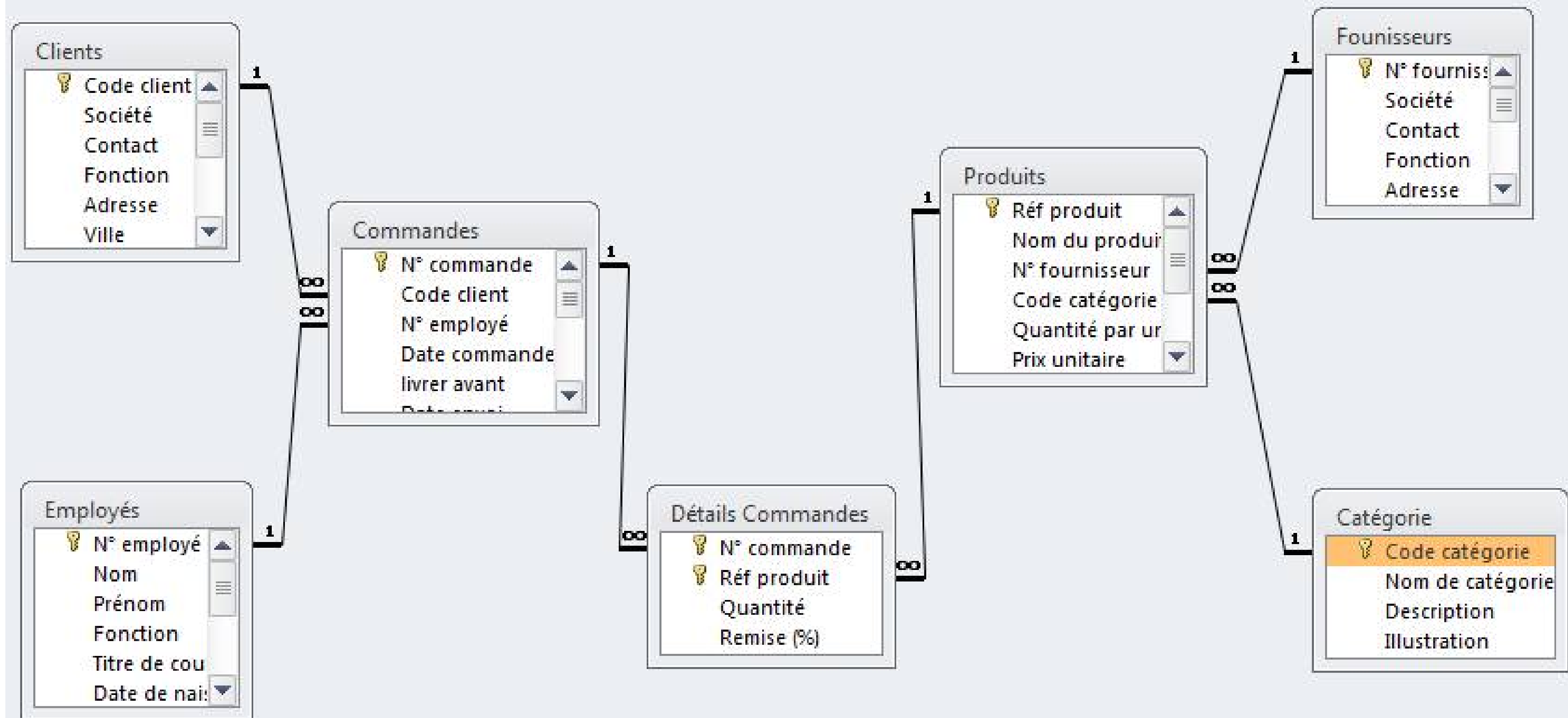
Employé (<u>N° Employé</u>, nom, prénom, Fonction)

Commande (<u>N° commande</u>, Date, #N° client, #N° Employé)

Détail commande (<u>N° commande, N° produit</u>, PrixUnitaire, Quantité)

Produit (<u>N° produit</u>, Nom produit, prix, #N° Fournisseur)

Fournisseur (<u>N° fournisseur</u>, société, contact, Fonction)


Clients
Code client
Société
Contact
Fonction
Adresse
Ville
Commandes
N° commande
Code client
N° employé
Date commande
livrer avant
Employés
N° employé
Nom
Prénom
Fonction
Titre de cou
Date de nai:
Détails Commandes
N° commande
Réf produit
Quantité
Remise (%)
Produits
Réf produit
Nom du produi
N° fournisseur
Code catégorie
Quantité par ur
Prix unitaire
Founisseurs
N° fourniss
Société
Contact
Fonction
Adresse
Catégorie
Code catégorie
Nom de catégorie
Description
Illustration
1
∞

Normalisation du (MLDR)

Pour quoi ?

NORMALISATION

Eviter les anomalies. tels que les anomalies de lecture, les anomalies d'écriture, la redondance des données et la contre performance.

Permet de vérifier la robustesse de la conception pour améliorer la modélisation

faciliter la mémorisation des données en évitant la redondance et les problèmes sous-jacents de mise à jour ou de cohérence

Dépendance Fonctionnelle (DF)

Permet de passer d'un ensemble de propriétés non structuré à un modèle conceptuel des données formé d'entités et d'associations et au modèle relationnel correspondant.

On dit que **b** est en dépendance fonctionnelle (DF) de **a** si à une valeur quelconque de la propriété **a**, on ne peut faire correspondre qu'une seule valeur au plus de la propriété **b**.

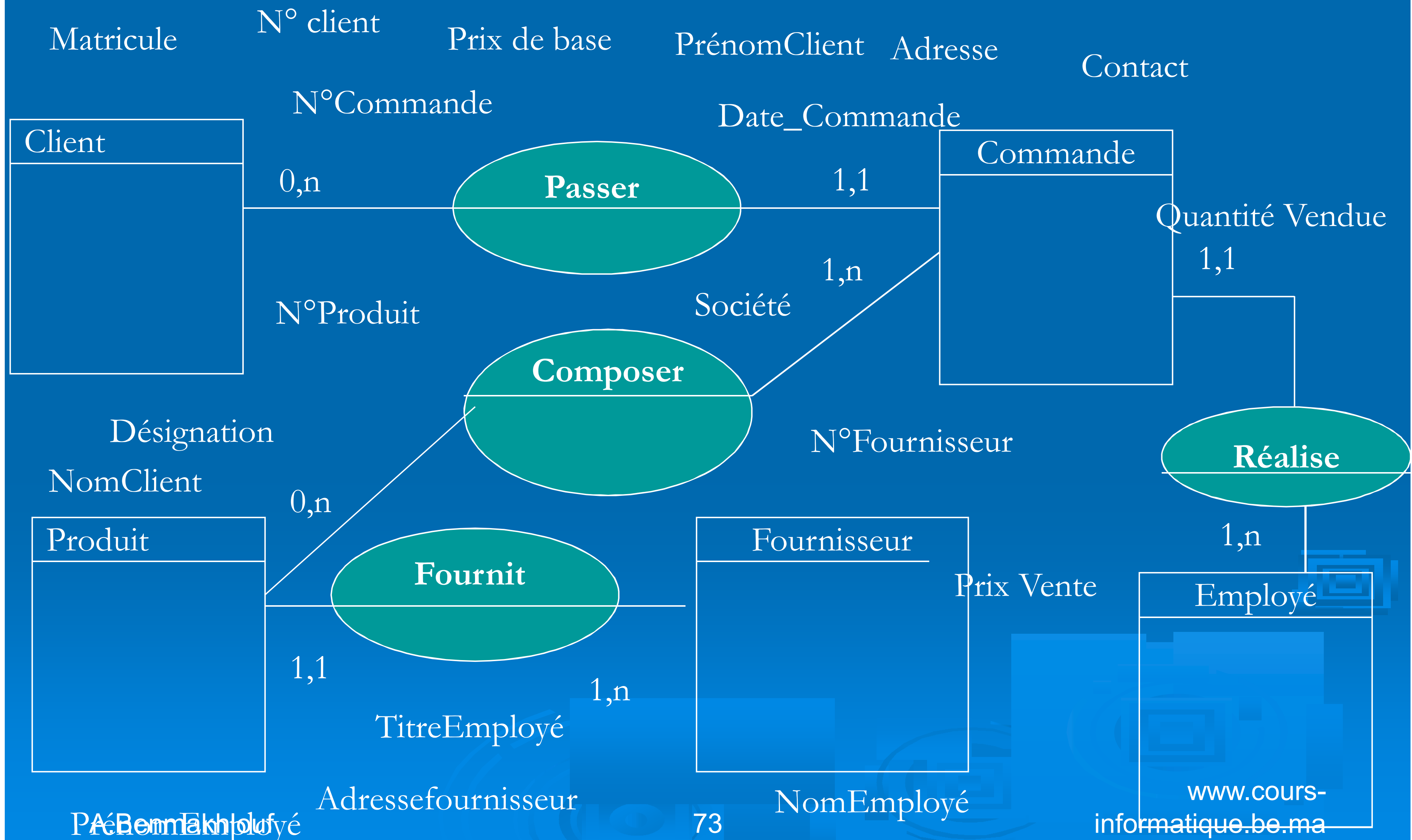
On note a → b
(source) → (cible)

Exemple :
Num client → Nom client

La réciproque est fausse :
Nom client ↛ Num client

Normalisation du (MLDR)

Dépendance Fonctionnelle (DF)



www.cours-informatique.be.ma

DF à partir de propriétés concaténées

Il peut exister des dépendances fonctionnelles à partir de propriétés concaténées (DFC),

Exemple : Considérons une commande qui comporte plusieurs produits

Num_Commande + Ref_Produit ⟶ **quantité commandée**

Les dépendances fonctionnelles dont la source est formée de plusieurs propriétés doivent être élémentaires, c'est-à-dire ne pas être créées artificiellement.

Exemple :

Num Commande + Num Client ↛ **date commande**

n'est pas une DF élémentaire car on n'a pas besoin du numéro de client pour connaître la date de commande

Propriétés des dépendances fonctionnelles

- **UNION**

Si on a deux DF ayant la même source, on peut les rassembler en une seule, en séparant les cibles par une virgule.

Si a → b et a → c alors ont peut écrire que a → b, c

Exemple :

Référence → Désignation
Référence →Prix de vente unitaire

Alors par union on a :

Référence → Désignation, Prix de vente unitaire

Propriétés des dépendances fonctionnelles

- **TRANSITIVITE**

Si a → b et b → c alors on a a → c

Exemple : Num Médecin → Code Service
Code Service → Num Hopital
Alors on a Num Médecin → Num Hopital

Les DF qui peuvent être déduites par transitivité de deux autres DF (qui ne sont pas directes) doivent être éliminées car elles sont alors **redondantes**.

Il ne reste alors que les DF directes, c'est-à-dire celles qui ne peuvent pas être retrouvées par transitivité.

➢ 1ère forme Normale

Une relation est normalisée en première forme normale si :

1) elle possède une clé identifiant de manière unique et stable chaque ligne

2) chaque attribut est <u>monovalué</u> (ne peut avoir qu'une seule valeur par ligne)

3) aucun attribut n'est <u>décomposable</u> en plusieurs attributs significatifs

Contre-exemple :

EMPLOYE(Nom, Enfants, Diplômes)

NOM	DIPLOME		ENFANTS	
KARIMI	Nature	Année	Prénom	Année de Naissance
	Bac	1975	Ahmed	1988
	Licence	1980	Sara	1992
			Anas	2000

➢ 2ème forme Normale

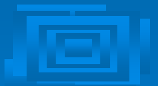
Une relation R est en deuxième forme normale si et seulement si :

- **elle est en 1FN**
- **tout attribut non clé est totalement dépendant de toute la clé.**

Autrement dit, aucun des attributs ne dépend que d'une partie de la clé.

La 2FN n'est à vérifier que pour les relations ayant une clé composée. Une relation en 1FN n'ayant qu'un seul attribut clé est toujours en 2FN

Contre-exemple :

LIGNE_COMMANDE(Num_cde, RéférenceProd, DésignationProd, Quantité) 

Les 3 formes normales

➢ 3ème forme Normale

Une relation est en 3° forme normale si et seulement si :

➢ elle est en 2° forme normale

➢ et tout attribut doit dépendre directement de la clé, c'est-à-dire qu'aucun attribut ne doit dépendre de la clé par transitivité.

Autrement dit, aucun attribut ne doit dépendre d'un autre attribut non clé.

Contre-exemple :

CLIENT(Num_client, Nom_client, code_categ, nom_categ)

Résumé

Modèle normalisé = relations (table) avec

- **une clé, qui permet de distinguer chaque occurrence**
- **des attributs élémentaires (1FN)**
- **en dépendance de TOUTE la clé (2FN),**
- **et RIEN QUE de la clé (3FN)**

Exercices

- Description des commandes d'un client

Commande (N°Commande, Date_commande, Date_livraison, Code_client, nom client, adresse)

Avec les dépendances fonctionnelles suivantes:

N°Commande -→ Date_commande, Date_livraison
Code client --→ nom client, adresse

- Détails des commandes

Détails_Commande (<u>N°Commande, N°Produit</u>, Quantité, Désignation-Prod)

Avec les dépendances fonctionnelles suivantes:

N°Commande, N°Produit -→ Quantité
N°Produit --→ Désignation_Prod

www.cours-informatique.be.ma

Modèle Physique de Donnée (MPD)

- C'est l'étape de la création (implantation) du MLD et de la manipulation des données en utilisant les langages LDD et LMD
- Dans cette étape en utilise un SGBD
- Le SGBD utilisé est MS-Access

Exemple : implantation de la BDD , « Magasin de Vente de Produit » dans Access

L'algèbre relationnelle est une théorie mathématique qui définit des opérations qui peuvent être effectuées sur des relations (Tables).

Le formalisme de L'algèbre relationnelle est au cœur du langage de requête de SQL.

Les principes de l'algèbre relationnelle sont beaucoup utilisés de nos jours par les SGBD pour la gestion des bases de données informatiques comme le SQLserver, DBase, Access, etc.

Nous pouvons distinguer trois familles d'opérateurs relationnels :

- **Les opérateurs unaires** (la sélection et la projection), permettent de produire une nouvelle table à partir d'une autre table.

- **Les opérateurs binaires** ensemblistes (l'union, l'intersection et la différence) permettent de produire une nouvelle relation à partir de deux relations de même degré et de même domaine.

- **Les opérateurs binaires ou n-aires** (le produit cartésien, la jointure et la division) permettent de produire une nouvelle table à partir de deux ou plusieurs autres

la Sélection

La sélection génère une relation regroupant exclusivement touts les attributs de la relation R en vérifiant une expression logique « E ».

$$\sigma_{(E)} R$$

Produit		
RefProduit	NomProduit	Prix
3	Produit1	34
12	Produit2	2
34	Produit3	21
5	Produit4	10

$$\sigma_{(Prix \geq 20)} Produit$$

Produit		
RefProduit	NomProduit	Prix
3	Produit1	34
34	Produit3	21

Projection

La projection consiste à supprimer les attributs autres que (A1, A2,.., An) d'une relation et à éliminer les occurrences en double apparaissant dans la nouvelle version.

$$\prod_{(A_1,A_2,\ldots\ldots,A_n)} R$$

Produit		
RefProduit	NomProduit	Prix
3	Produit1	34
12	Produit2	2
34	Produit3	21
5	Produit4	10

$$\prod_{(RefProduit,NomProduit)} produit$$

RefProduit	NomProduit
3	Produit1
12	Produit2
34	Produit3
5	Produit4

$$\prod_{(RefProduit,NomProduit)} \sigma_{(Prix \geq 20)} Produit$$

RefProduit	NomProduit
3	Produit1
34	Produit3

Union

une opération portant sur deux relations R1 et R2 ayant le même schéma (même attributs) et construisant une troisième relation constituée des enregistrements appartenant à l'une ou l'autre des deux relations R1 et R2 sans doublon.

$$R_1 \cup R_2$$

R1

RefProduit	NomProduit
3	Produit1
12	Produit2
34	Produit3
5	Produit4

R2

RefProduit	NomProduit
78	Produit5
12	Produit2
134	Produit6

$R_1 \cup R_2$

RefProduit	NomProduit
3	Produit1
12	Produit2
34	Produit3
5	Produit4
78	Produit5
134	Produit6

Intersection

une opération portant sur deux relations R1 et R2 ayant le même schéma et construisant une troisième relation dont les enregistrements sont constitués de ceux appartenant aux deux relations

$$R_1 \cap R_2$$

R1	
RefProduit	NomProduit
3	Produit1
12	Produit2
34	Produit3
5	Produit4

R2	
RefProduit	NomProduit
3	Produit1
12	Produit2
45	Produit5
675	Produit6

$R_1 \cap R_2$	
RefProduit	NomProduit
3	Produit1
12	Produit2

Produit cartésien

Le *produit cartésien* est une opération portant sur deux relations R1 et R2 et qui construit une troisième relation regroupant exclusivement toutes les possibilités de combinaison des occurrences des relations R1 et R2

$$R_1 \times R_2$$

R1	
A	**B**
a	Z1
b	Z2

R2	
C	**D**
45	P1
32	P2
12	P3

$R_1 \times R_2$			
A	**B**	**C**	**D**
a	Z1	45	P1
a	Z1	32	P2
a	Z1	12	P3
b	Z2	45	P1
b	Z2	32	P2
b	Z2	12	P3

Jointure

opération portant sur deux relations R1 et R2 qui construit une troisième relation regroupant exclusivement toutes les possibilités de combinaison des occurrences des relations R1 et R2 qui satisfont l'expression logique E. Il s'agit d'une opération binaire commutative.

Une jointure est donc une sélection sur un produit cartésien.

$$\sigma_{(E)}(R_1 \times R_2)$$

R1	
A	**B**
a	Z1
b	Z2

R2	
C	**D**
45	P1
32	P2
12	P3

$\sigma_{(C \geq 30\ ET\ B="Z1")}(R_1 \times R_2)$			
A	**B**	**C**	**D**
a	Z1	45	P1
a	Z1	32	P2

Thêta-jointure

une jointure dans laquelle l'expression logique E est une simple comparaison entre un attribut A1 de la relation R1 et un attribut A2 de la relation R2.

R1	
A	**B**
23	x
21	y
45	z

R2	
C	**D**
56	x
12	y
11	z

$\sigma_{(A \leq C)}(R_1 \times R_2)$			
A	**B**	**C**	**D**
23	x	56	x
21	y	56	x
45	z	56	x

Équi-jointure

C'est une thêta-jointure dans laquelle l'expression logique E est un test d'égalité entre un attribut A1 de la relation R1 et un attribut A2 de la relation R2.

R1	
A	B
23	x
21	y
45	z

R2	
C	D
56	x
12	y
11	z

$\sigma_{B=D}(R1 \times R2)$			
A	B	C	D
23	x	56	x
23	x	12	Y
23	x	11	Z
21	y	56	x
21	y	12	Y
21	y	11	Z
45	z	56	x
45	z	12	Y
45	z	11	Z

$\sigma_{B=D}(R1 \times R2)$			
A	B	C	D
23	x	56	x
21	y	12	y
45	z	11	z

SQL : Structured Query Language

SQL est un langage complet de gestion de bases de données relationnelles. C'est un:

- un langage d'interrogation de la base (ordre SELECT)
- un langage de manipulation des données (LMD) (ordres UPDATE, INSERT, DELETE)
- un langage de définition des données (LDD) (ordres CREATE, ALTER, DROP),
- un langage de contrôle de l'accès aux données (LCD) (ordres GRANT, REVOKE).

SQL : Structured Query Language

Le langage SQL est utilisé par les principaux SGBDR : Oracle, Access, SQL server, MySQL…

Chacun de ces SGBDR a cependant sa propre variante du langage.

On présente un noyau de commandes disponibles sur l'ensemble de ces SGBDR,

La commande **CREATE TABLE**

Permet de créer une table dans la base de données courante.

CREATE TABLE table (champ1 type propriété, champ2 type propriété,..);

Langage SQL (Définition de données) (LDD)

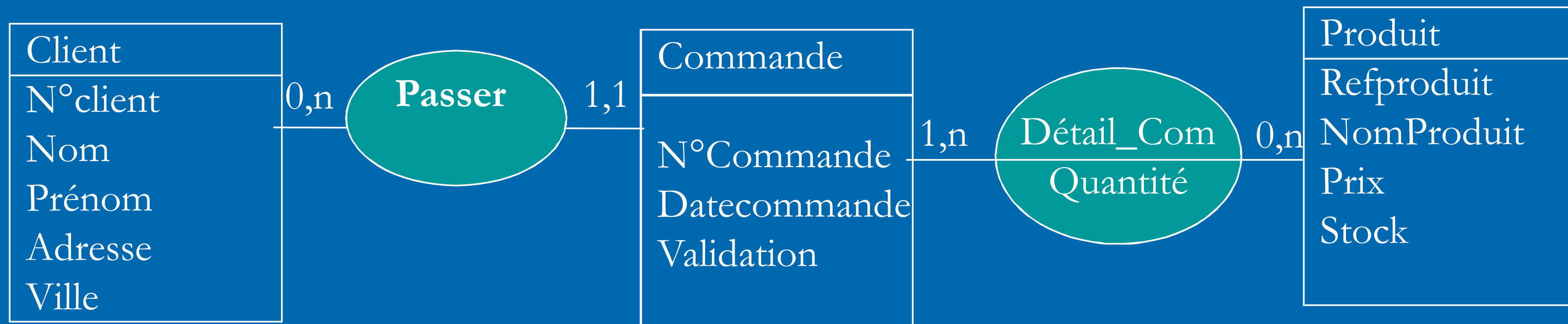
Type de Donnée SQL(Access)	Signification
INTEGER	Nombre entier
SINGLE	Nombre réel
DOUBLE	Nombre réel double
NUMERIC	Nombre réel double
BIT	Nombre Boolean
COUNTER	NuméroAuto
CURRENCY, MONEY	Monétaire
DATE, TIME, DATETIME	Date
VARCHAR	Texte avec max 255 caractères
CHAR(n), TEXT(n)	Texte avec max n caractères
LONGTEXT	Texte memo max 32Ko
LONGBINARY	Objet OLE

Langage SQL (Définition de données) (LDD)

Propriété de Donnée SQL(Access)	Signification
`NULL ou NOT NULL`	autorise ou non que le champ puisse être vide.
`UNIQUE`	indique que deux enregistrements ne pourront avoir la même valeur dans ce champ.
`PRIMARY KEY`	indique que ce champ est la clef primaire
`REFERENCES`	Implique une intégrité référentielle

```
CREATE TABLE Table1 (a1 TEXT PRIMARY KEY, b1 TEXT(4) NOT NULL, c1 INT);

CREATE TABLE Table2 (a2 TEXT PRIMARY KEY, b2 TEXT(4) NOT NULL, c2 TEXT
REFERENCES Table1(a1));
```



Client(N°client, Nom, Prénom, Adresse, Ville)
Produit(RefProduit, NomProduit, Prix, Stock)
Commande(N°Commande, DateCommande, Validation, #N°Client)
Détail_Com(N°Commande, Refproduit, Quantité)

Langage SQL (Définition de données) (LDD)

Client(N°client, Nom, Prénom, Adresse, Ville)

```
CREATE TABLE Client (N°Client TEXT PRIMARY KEY, Nom TEXT NOT NULL,
Prénom TEXT, Adresse TEXT, Ville Text);
```

Produit(RefProduit, NomProduit, Prix, Stock)

```
CREATE TABLE Poduit (RefProduit TEXT(4) PRIMARY KEY, NomProduit
TEXT NOT NULL, Prix FLOAT, Stock INTEGER);
```

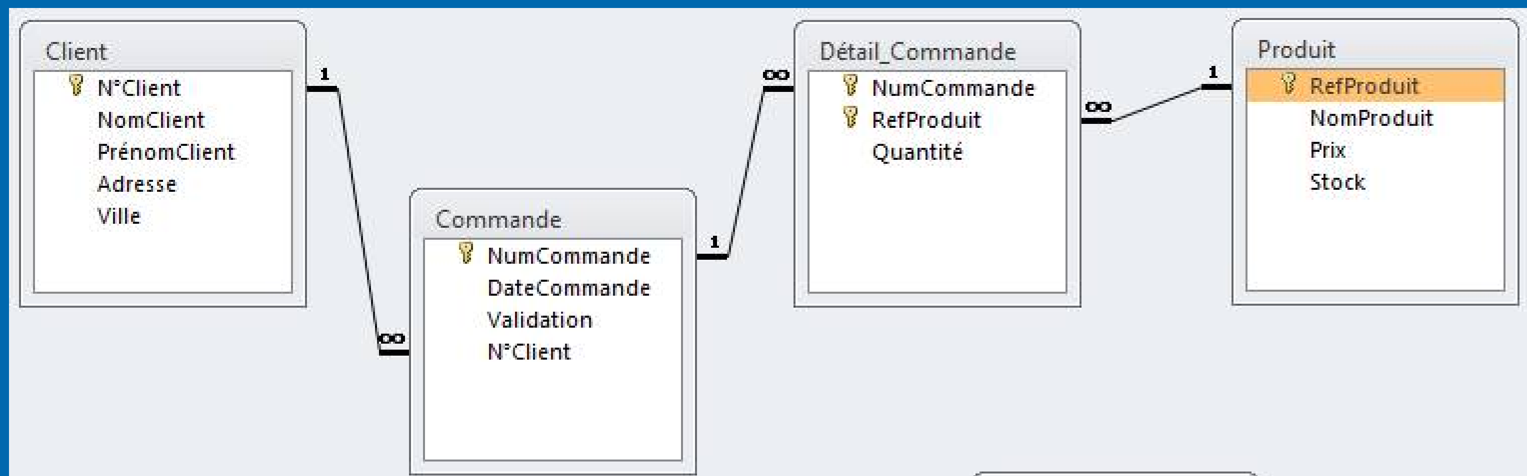
Commande(N°Commande, DateCommande, Validation, #N°Client)

```
CREATE TABLE Commande (NumCommande COUNTER PRIMARY KEY,
DateCommande Date NOT NULL, Validation BIT,
N°Client TEXT REFERENCES Client(N°Client));
```

Détail_Com(N°Commande, Refproduit, Quantité)

```
CREATE TABLE Détail_Commande (NumCommande INT REFERENCES
Commande(NumCommande),
RefProduit TEXT(4) REFERENCES Produit(RefProduit), Quantité INT NOT NULL,
PRIMARY KEY (NumCommande, RefProduit));
```

Langage SQL (Définition de données) (LDD)



La commande **ALTER TABLE**

Permet de modifier la structure d'une table

ALTER TABLE table Action (spécifications du champ);

ADD : Ajoute un champ a une table
DROP : Supprime un champ d'une table
MODIFY : Modifie les caractéristiques d'un champ

Exemple

```
ALTER TABLE Client  ADD CodePostal TEXT NOT NULL;
```

La commande **INSERT**

La commande INSERT est utilisée pour ajouter des enregistrements ou des parties d'enregistrements dans des tables.

```
INSERT INTO table (champ1,champ2,...)
VALUES ('valeur1','valeur2',...);
```

INSERT INTO détailcommande (N°Commande, N°Produit, Quantité, PrixVente)
VALUES (113, 1, 2, 25), (113, 2, 7, 25), (115, 4, 2, 25), (114, 2, 3, 25), (113, 5, 8, 25)

La commande **UPDATE**

changer des valeurs dans des champs d'une table

```
UPDATE table
SET champ1 = nouvelle_valeur1, champ2 = nouvelle_valeur2,
champ3 = nouvelle_valeur3
WHERE condition;
```

```
UPDATE Produit SET Prix = Prix*(1+0.05), Stock=Stock+6
WHERE RefProduit=''A210'';
```

```
UPDATE Produit SET [Prix unitaire] = [Prix unitaire]*(1-0.1)
WHERE [N°fournisseur]=7;
```

```
UPDATE Produit SET [Prix unitaire] = [Prix unitaire]*(1-0.1)
WHERE [N°fournisseur]=7 AND Prix < 230;
```

La commande SELECT

La commande SELECT est la commande la plus complexe de SQL

Permet de faire des requêtes pour récupérer des données dans les tables.

Plusieurs Genre de Requêtes

- Requêtes simples
- Requêtes simples avec des critères
- Requêtes simples avec calcul sur les dates
- Requêtes simples avec création de nouveaux attributs
- Requêtes multi-tables avec regroupement et avec une synthèse

La commande SELECT

Requêtes simples

```
SELECT `champ1 `, ` champ2 `, ...
FROM table1;
```

$$\prod_{(champ1,champ2,....)} table1$$

Exemple

```
SELECT CodeClient, NomClient, AdresseClient
FROM Client
```

```
SELECT *
FROM Client
```

La commande SELECT

Requêtes simples avec critères

WHERE exp1 = exp2
WHERE exp1 != exp2
WHERE exp1 < exp2
WHERE exp1 > exp2
WHERE exp1 <= exp2
WHERE exp1 >= exp2
WHERE exp1 LIKE exp2
WHERE exp IS NULL
WHERE exp IS NOT NULL

La commande SELECT

Requêtes simples avec critères

```
SELECT  champ1, champ2, ....
FROM Table1
WHERE condition;
```

$$\prod_{(champ1,champ2,....)} (\sigma_{condition} table1)$$

<u>Exemple</u>

```
SELECT RefProduit, Désignation, PrixUnitaire
FROM Produit
WHERE PrixUnitaire <=120
```

Utilisation des Opérateurs **AND** et **OR** pour combiner plusieurs critères

Langage SQL (Interrogation de données) (LID)

La commande SELECT

Requêtes simples avec critères

Pour classer les valeurs d'un champ on utilise l'opérateur **ORDER BY**

```
SELECT  champ1, champ2, ....
FROM Table
WHERE condition
ORDER BY champ1
```

```
SELECT  champ1, champ2, ....
FROM Table
WHERE condition
ORDER BY champ1  DESC
```

descending

Exemple

```
SELECT RefProduit, Désignation, PrixUnitaire
FROM Produit
ORDER BY PrixUnitaire DESC
```

La commande SELECT

Requêtes simples avec création de nouveaux attributs

Par fois nous avons besoin de calculer des données à partir de données brutes

```
SELECT champ1, formule AS Nouveau_champ
FROM table ;
```

Exemple

```
SELECT [nom du produit], PUHT*(1+TVA) AS PTTC
FROM Produit
```

La commande SELECT

Requêtes simples avec calcul sur les dates

Quelques fonctions dates

- **DAY(Date)**: Renvoie le jour de la date
- **MONTH(Date)**: Renvoie le mois de la date
- **YEAR(Date)** : Renvoie l'année de la Date
- **WEEKDAY(Date)**: Renvoie le jour de la semaine d'une date
- **NOW()** : Renvoie la date actuelle (date du système)

La commande SELECT

Requêtes simples avec calcul sur les dates

Exemple

```
SELECT  N°Commande, DateCommande
FROM Commande
WHERE  YEAR(DateCommande) = YEAR(NOW())
```

La commande SELECT

Requêtes MutiTables (Les Jointures)

Les jointures (Equi-jointure) permettent de sélectionner des informations dans plusieurs tables grâce aux relations existant entre ces tables.

- Les Jointures Internes
- Les Jointures Externes

Langage SQL (Interrogation de données) (LID)

La commande SELECT

Requêtes MutiTables (Les Jointures)

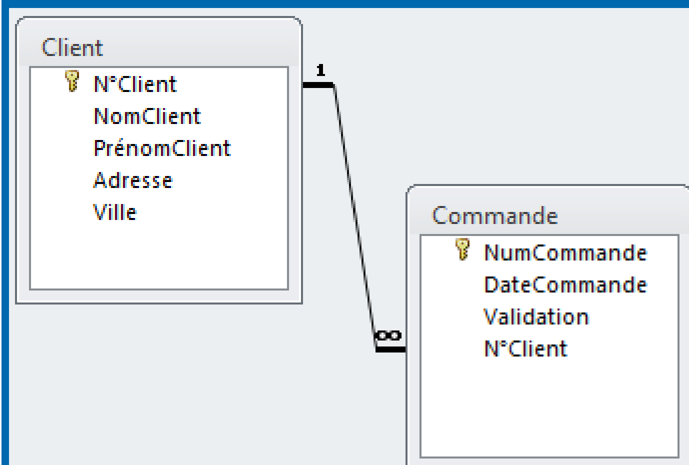
- Les Jointures Internes

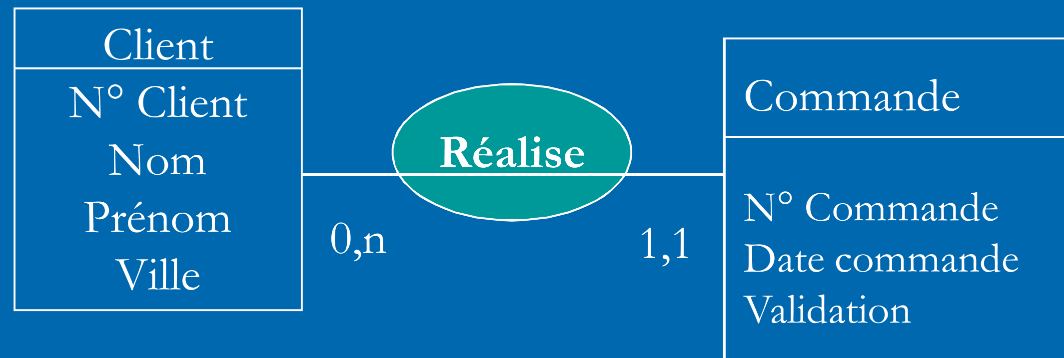
R1 (a1, b1, c1, d1)
R2 (a2, b2, c2, a1)

```
SELECT R1.a1, b1, b2, c2
FROM R1, R2
WHERE R1.a1 = R2.a1
```

```
SELECT R1.a1, b1, b2, c2
FROM R1 INNER JOIN R2
ON R1.a1 = R2.a1
```

Requêtes MutiTables (Les Jointures Internes)





N°Client	Nom	Prénom	Ville
CL1	Nom1	Prénom1	Ville1
CL2	Nom2	Prénom2	Ville2
CL3	Nom3	Prénom3	Ville3
CL4	Nom4	Prénom4	Ville4

N°Com	DateCom	Validation	N°Client
C1	12/03/2014	OUI	CL1
C2	12/04/2014	OUI	CL2
C3	12/05/2014	OUI	CL2
C4	12/05/2014	OUI	CL2

SELECT Client.N°Client, Nom, N°Com, DateCom
FROM Client *INNER JOIN* Commande
ON Client.N°Client = Commande.N°Client

N°Client	Nom	N°Com	DateCom
CL1	Nom1	C1	12/03/2014
CL2	Nom2	C2	12/04/2014
CL2	Nom2	C3	12/05/2014
CL2	Nom2	C4	12/05/2014

Les clients qui n'ont pas réalisé de commande ne figureront pas dans la liste

Requêtes MutiTables (Les Jointures Externes)

Des jointures qui permettent d'afficher tous les occurrences de la table qui se trouve à droite de la jointure (à gauche de la jointure) tout en vérifiant l'égalité entre les attributs joint.

R1 (<u>a1</u>, b1, c1, d1)
R2 (a2, b2, c2, a1)

SELECT R1.a1, b1, b2, c2
FROM R1 ***LEFT*** *JOIN* R2
ON R1.a1 = R2.a1

SELECT R1.a1, b1, b2, c2
FROM R1 ***RIGHT*** *JOIN* R2
ON R1.a1 = R2.a1

N°Client	Nom	Prénom	Ville
CL1	Nom1	Prénom1	Ville1
CL2	Nom2	Prénom2	Ville2
CL3	Nom3	Prénom3	Ville3
CL4	Nom4	Prénom4	Ville4

N°Com	DateCom	Validation	N°Client
C1	12/03/2014	OUI	CL1
C2	12/04/2014	OUI	CL2
C3	12/05/2014	OUI	CL2
C4	12/05/2014	OUI	CL2

SELECT Client.N°Client, Nom, N°Com, DateCom
FROM Client *LEFT JOIN* Commande
ON Client.N°Client = Commande.N°Client

N°Client	Nom	N°Com	DateCom
CL1	Nom1	C1	12/03/2014
CL2	Nom2	C2	12/04/2014
CL2	Nom2	C3	12/05/2014
CL2	Nom2	C4	12/05/2014
CL3	Nom3		
CL4	Nom4		

La commande SELECT

Requêtes MutiTables avec Regroupement et synthèse

SELECT Client.N°Client, Nom, **COUNT(**N°Com**)**
FROM Client *LEFT JOIN* Commande
ON Client.N°Client = Commande.N°Client
GROUP BY Client.N°Client, Nom

N°Client	Nom	Nbre
CL1	Nom1	1
CL2	Nom2	3
CL3	Nom3	0
CL4	Nom4	0



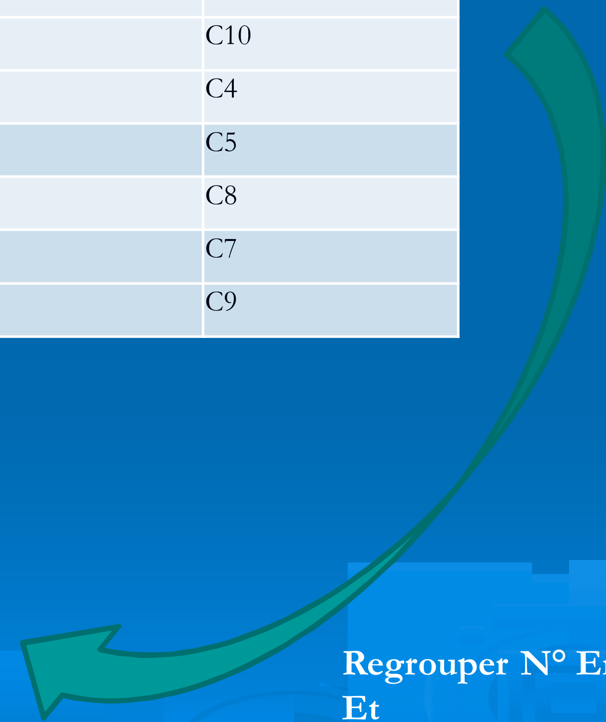
N°Client	Nom	N°Com
CL1	Nom1	C1
CL2	Nom2	C2
CL2	Nom2	C3
CL2	Nom2	C4
CL3	Nom3	
CL4	Nom4	

Langage SQL (Interrogation de données) (LID)

N°Employé	N°Commande
E1	C1
E2	C2
E1	C3
E3	C4
E4	C5
E2	C6
E5	C7
E4	C8
E5	C9
E2	C10



N°Employé	N°Commande
E1	C1
E1	C3
E2	C2
E2	C6
E2	C10
E3	C4
E4	C5
E4	C8
E5	C7
E5	C9



N°Employé	Nbre Commande
E1	2
E2	3
E3	1
E4	2
E5	2

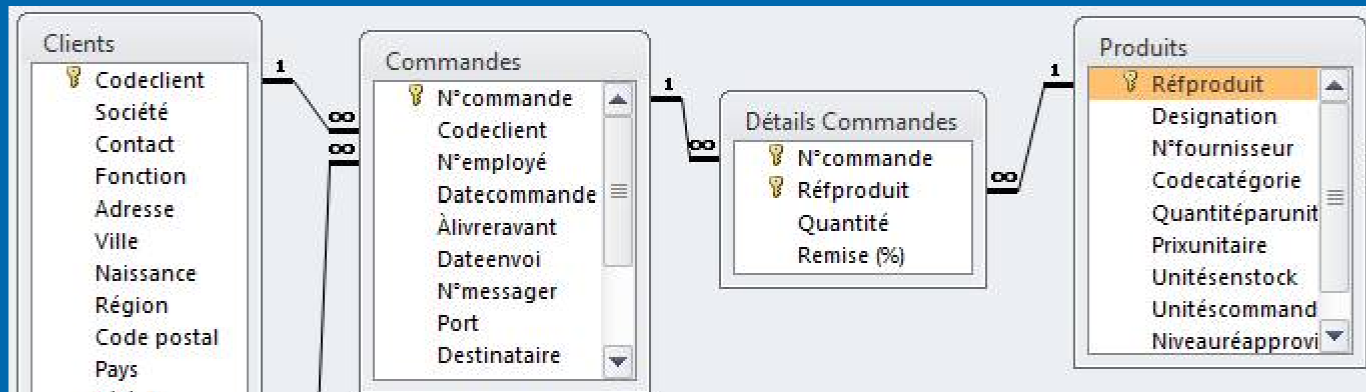
Regrouper N° Employé
Et
Compter N° Commande

Langage SQL (Interrogation de données) (LID)

Requête Multi-Table contenant plusieurs jointures

(((**Table1** *Jointure* **Table2**) *Jointure* **Table3**) *Jointure* **Table4**)…….

Langage SQL (Interrogation de données) (LID)



Montant total par Client

SELECT Clients.codeclient, société, SUM(Prixunitaire*Quantité) AS CA

FROM (((Clients INNER JOIN Commandes

ON Clients.codeclient=Commandes.codeclient)

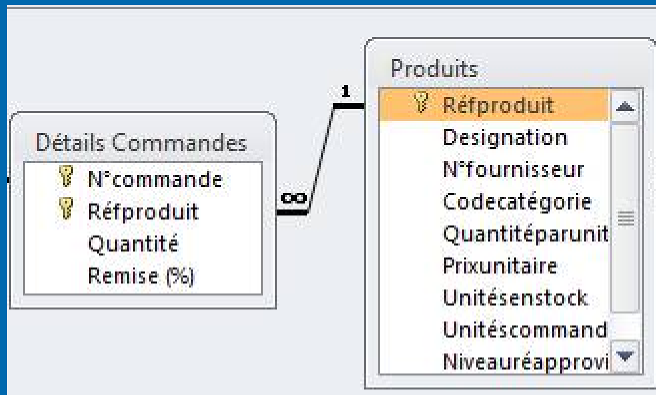
INNER JOIN [Détails Commandes]

ON Commandes.N°commande=[Détails Commandes].N°commande)

INNER JOIN Produits

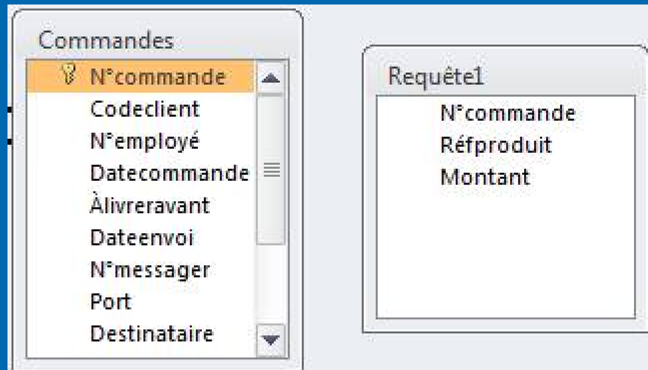
ON [Détails Commandes].Réfproduit=Produits.Réfproduit)

GROUP BY Clients.codeclient, société;



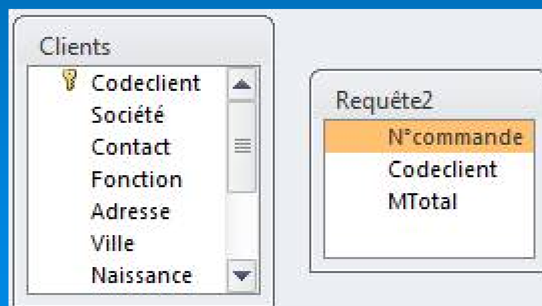
Requête1: Montant/Commande/Produit

SELECT [Détails Commandes].N°commande, [Détails Commandes].Réfproduit, Quantité*Prixunitaire AS Montant
FROM [Détails Commandes] INNER JOIN Produits ON [DétailsCommandes].Réfproduit=Produits.Réfproduit;



Requête2: Montant/Commande/Client

SELECT Commandes.N°commande, codeclient, SUM(Montant) AS MTotal
FROM Commandes INNER JOIN Requête1
ON Commandes.N°commande = Requête1.N°commande
GROUP BY codeclient, Commandes.N°commande;



Requête3: Montant/Client

SELECT Clients.Codeclient, société, SUM(Mtotal) AS CA
FROM Clients INNER JOIN Requête2 ON Clients.Codeclient=[Requête2].Codeclient
GROUP BY Clients.Codeclient, société;

Sous Requêtes

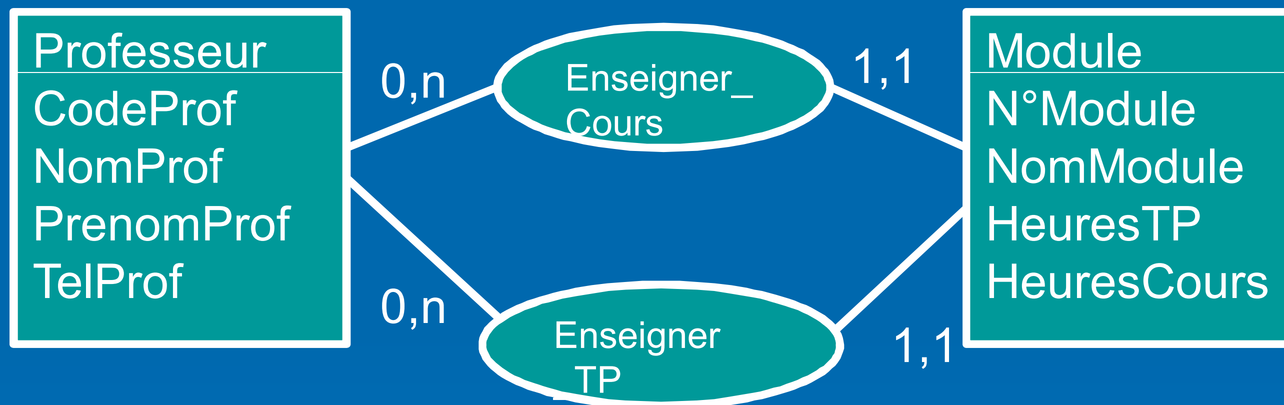
On peut imbriquer autant de requêtes que l'on veut. La condition après la clause WHERE peut porter sur le résultat d'une autre requête (ou sous-requête).

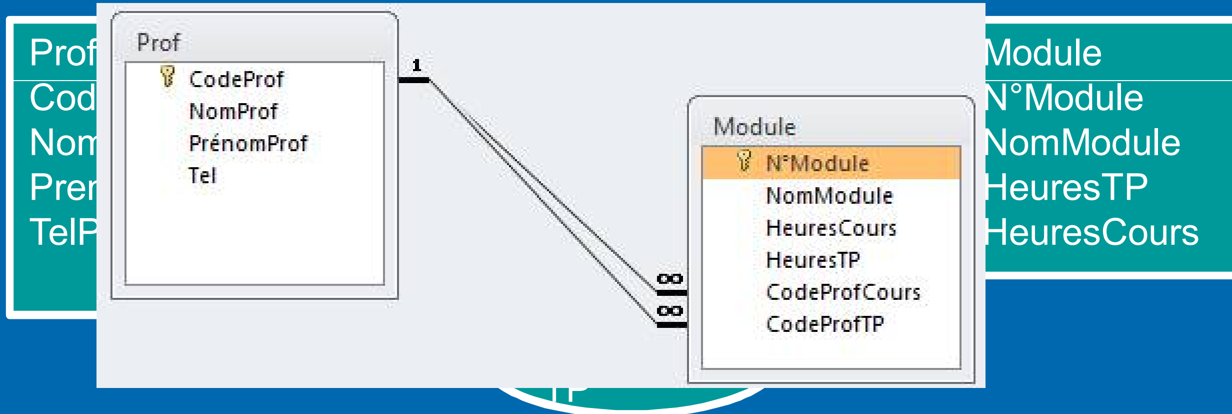
```
SELECT Réfproduit, Designation, Prixunitaire
FROM Produits
WHERE Prixunitaire > (SELECT AVG(Prixunitaire) FROM Produits);
```

L'UNION

Lorsqu'on veut que les résultats de plusieurs requêtes soient combinés entre eux, on utilise la clause UNION. UNION va fusionner les résultats des requêtes.

```
SELECT Champ1, Champ2 FROM Table1
UNION SELECT Champ1, Champ2 FROM Table2;
```





HeuresTP

SELECT CodeProf, NomProf,
SUM(HeuresTP) AS HTotal
FROM Prof INNER JOIN [Module]
ON Prof.CodeProf=Module.CodeProfTP
GROUP BY CodeProf, NomProf;

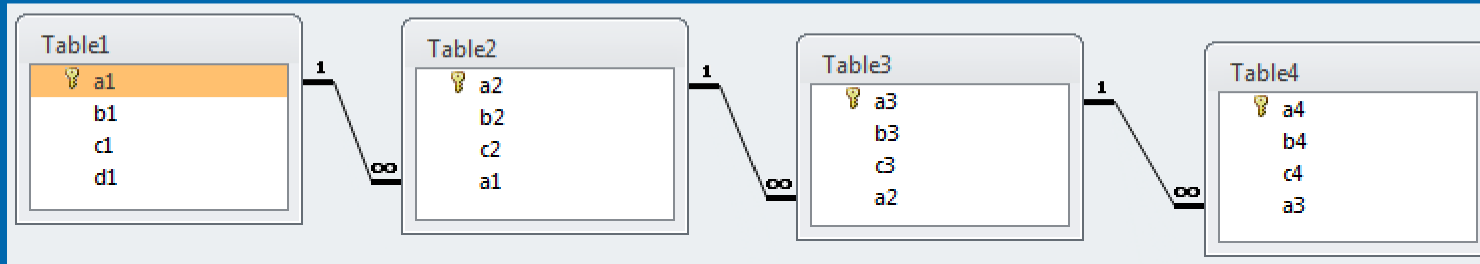
HeuresCours

SELECT CodeProf, NomProf,
Sum(HeuresCours) AS HTotal
FROM Prof INNER JOIN [Module]
ON Prof.CodeProf = Module.CodeProfCours
GROUP BY CodeProf, NomProf;

HeuresTotal

SELECT CodeProf, NomProf, SUM(HTotal) AS HCoursTP
FROM **(SELECT* FROM HeuresCours UNION SELECT* FROM HeuresTP)**
GROUP BY CodeProf, NomProf;

La Suppression et la mise à jour en cascade



Dans le cas normal on ne peut pas supprimer un enregistrement ou mettre à jour la valeur de la clé de la table de ma table source qui se trouve dans la table cible.

On peut permettre une suppression ou une mise à jour en cascade

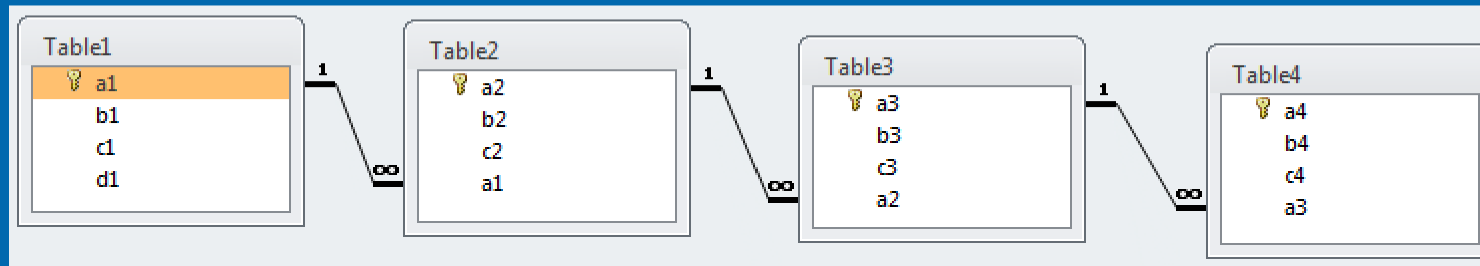
CREATE TABLE [Table1] (a1 Text PRIMARY Key, b1 Text, c1 Text, d1 Text)

CREATE TABLE [Table2] (a2 Text PRIMARY Key, b2 Text, c2 Text, a1 Text REFERENCES Table1(a1) ***ON DELETE CASCADE ON UPDATE CASCADE***)

CREATE TABLE [Table3] (a3 Text PRIMARY Key, b3 Text, c3 Text, a2 Text REFERENCES Table2(a2) ***ON DELETE CASCADE ON UPDATE CASCADE***)

CREATE TABLE [Table4] (a4 Text PRIMARY Key, b4 Text, c4 Text, a3 Text REFERENCES Table3(a3) ***ON DELETE CASCADE ON UPDATE CASCADE***)

La Suppression et la mise à jour en cascade



Modifier des relations

Table/Requête : Table1

Table/Requête liée : Table2

a1 | a1

OK

Annuler

Type jointure…

Nouvelle relation…

- [x] Appliquer l'intégrité référentielle
- [x] Mettre à jour en cascade les champs correspondants
- [x] Effacer en cascade les enregistrements correspondants

Type de relation : Un-à-plusieurs